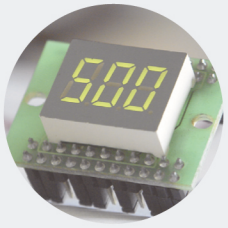
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES






ALGORITMO GENÉTICO PARA EMPACOTAMENTO DE CONTENTORES

RICARDO MORGADO PIRES
Novembro de 2012

POLITÉCNICO
DO PORTO

ALGORITMO GENÉTICO PARA EMPACOTAMENTO DE CONTENTORES

Ricardo Morgado Pires

Mestrado em Engenharia Eletrotécnica e de Computadores

Área de Especialização de Sistemas de Planeamento e Industrial

Departamento de Engenharia Eletrotécnica

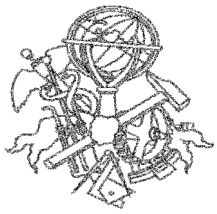
Instituto Superior de Engenharia do Porto

2012

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: Ricardo Morgado Pires, 1050802@isep.ipp.pt

Orientação científica: Cecília Maria do Rio Fernandes Moreira Reis, cmr@isep.ipp.pt



Mestrado em Engenharia Eletrotécnica e de Computadores

Área de Especialização de Sistemas de Planeamento e Industrial

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

4 de Novembro de 2012

Agradecimentos

Gostaria de agradecer a todos que direta ou indiretamente me ajudaram a criar e a desenvolver este trabalho:

- Em primeiro lugar quero agradecer à Prof. Cecília Maria do Rio Fernandes Moreira Reis, minha orientadora, que me deu todo o apoio para desenvolver este trabalho.

- Aos meus Amigos, por o serem e pelas sugestões que me derem para este trabalho.

- E finalmente à minha namorada Alexandra Fradique que sempre me apoiou, nos bons, menos bons e sobretudo nos maus momentos.

Resumo

No âmbito da investigação operacional o problema de empacotamento de contentores é conhecido por procurar definir uma configuração de carga, de forma a otimizar a utilização de um espaço disponível para efetuar o empacotamento.

Este problema pode ser apresentado em diversas formas, formas estas que variam em função das características de cada empacotamento. Estas características podem ser: o tipo de carga que se pretende carregar (homogénea ou heterogénea), a possibilidade de a carga poder sofrer rotações em todas as suas dimensões ou apenas em algumas, o lucro que está associado a cada caixa carregada ou restrições inerentes ao contentor como por exemplo dimensões.

O interesse pelo estudo de problemas de empacotamento de contentores tem vindo a receber cada vez mais ênfase por várias razões, uma delas é o interesse financeiro dado que o transporte é uma prática que representa custos, sendo importante diminuir estes custos aproveitando o volume do contentor da melhor forma. Outra preocupação que motiva o estudo deste problema prende-se com fatores ambientes, onde se procura racionalizar os recursos naturais estando esta também ligada a questões financeiras.

Na literatura podem ser encontradas varias propostas para solucionar este problema, cada uma destas dirigidas a uma variante do problema, estas propostas podem ser determinísticas ou não determinísticas onde utilizam heurísticas ou metaheurísticas.

O estudo realizado nesta dissertação descreve algumas destas propostas, nomeadamente as metaheurísticas que são utilizadas na resolução deste problema.

O trabalho aqui apresentado traz também uma nova metaheurísticas, mais precisamente um algoritmo genético que terá como objetivo, apresentar uma configuração de carga para um problema de empacotamento de um contentor.

O algoritmo genético tem como objetivo a resolução do seguinte problema: empacotar várias caixas retangulares com diversos tamanhos num contentor. Este problema é conhecido como *Bin-Packing*.

A novidade que este algoritmo genético vai introduzir nas diversas solução apresentadas até à data, é uma nova forma de criar padrões iniciais, ou seja, é utilizada a heurística HSSI (Heurística de Suavização de Superfícies Irregulares) que tem como objetivo criar uma população inicial de forma a otimizar o algoritmo genético.

A heurística HSSI tenta resolver problemas de empacotamento simulando, o comportamento da maioria das pessoas ao fazer este processo na vida real, contudo, tem um campo de busca reduzido entre as soluções possíveis e será então utilizado um algoritmo genético para ampliar este campo de busca e explorar novas soluções.

No final pretende-se obter um *software* onde será possível configurar um dado problema de empacotamento de um contentor e obter, a solução do mesmo através do algoritmo genético.

Assim sendo, o estudo realizado tem como principal objetivo contribuir com pesquisas e conclusões, sobre este problema e trazer uma nova proposta de solução para o problema de empacotamento de contentores.

Palavras-Chave

Problema de Empacotamento de Contentor, Heurísticas, Algoritmo Genético.

Abstract

As part of the operational research the problem of load packing containers is known to try to define a configuration load, to optimize the use of available space to make the packaging.

This problem can be presented in several forms, these forms could be according to the characteristics of each packaging. These characteristics may include: the type of load which is intended load (homogeneous or heterogeneous), the possibility of the load can undergo rotations in all its forms or in only some, the profit is associated with each box loaded or restrictions inherent such as container dimensions.

Interest in the study of container packing problems has received increasing emphasis for several reasons, one of which is the financial interest given that transportation is a practice that represents costs being important to reduce these costs by taking advantage of the volume of container best. Another concern that motivates the study of this problem is related to environmental factors, which seeks to rationalize natural resources this also being linked to financial issues.

Can be found in the literature several proposals to solve this problem, each of these led to a variant of the problem, these proposals can be deterministic or nondeterministic where use heuristics or metaheurísticas.

The study in this thesis describes some of these proposals, particularly the metaheurísticas that are used to solving this problem.

The work presented here also brings a new metaheurísticas, specifically a genetic algorithm that will aim to present a load configuration for a problem of packing a container.

The genetic algorithm aims at solving the following problem: packing rectangular boxes with several different sizes in a container. This problem is known as Bin-Packing. The novelty of this genetic algorithm will introduce the various solution presented to date, is a new way to create initial patterns, HSSI is used heuristics (Heuristics Smoothing Irregular surfaces) that aims to create an initial population of order to optimize the genetic algorithm.

The HSSI heuristic tries to solve packing problems simulating the behavior of most people to do this process in real life, however, has a search field between small and possible solutions will be then used a genetic algorithm to extend the search field and explore new solutions.

At the end it is intended to obtain software where it is possible to configure a given problem of a packaging container and obtain the solution of the same through the genetic algorithm.

Thus, the study's main objective is to contribute to research and conclusions on this issue and bring a new proposal for a solution to the problem of packaging containers.

Keywords

Container Packaging Problem, Heuristics, Genetic Algorithm.

Résumé

Dans le cadre de la recherche opérationnelle au problème de récipients d'emballage est connu pour tenter de définir une configuration de charge, afin d'optimiser l'utilisation de l'espace disponible pour faire l'emballage.

Ce problème peut être présenté sous plusieurs formes, ces formes variant selon les caractéristiques de chaque emballage. Ces caractéristiques peuvent inclure: le type de charge qui a pour but de charge (homogène ou hétérogène), la possibilité de la charge peut subir des rotations dans toutes ses formes ou en partie seulement, le résultat est associé à chaque boîte de chargement ou restrictions inhérentes tels que les dimensions du récipient.

L'intérêt pour l'étude des problèmes d'empotage du conteneur a reçu une attention croissante pour plusieurs raisons, dont l'une est l'intérêt financier étant donné que le transport est une pratique qui représente des coûts importants étant de réduire ces coûts en profitant du volume du récipient mieux.

Une autre préoccupation qui motive l'étude de ce problème est liée à des facteurs environnementaux, qui visent à rationaliser les ressources naturelles de ce également liés aux questions financières.

On peut trouver dans la littérature plusieurs propositions pour résoudre ce problème, chacun d'entre eux conduit à une variante du problème, ces propositions peut être déterministes ou non déterministe où heuristiques d'utilisation ou méta heuristiques. L'étude de cette thèse décrit certaines de ces propositions, en particulier celle méta heuristiques sont utilisées dans la résolution de ce problème.

Le travail présenté ici apporte également un méta heuristique nouveau, en particulier un algorithme génétique qui aura pour but de présenter une configuration de charge pour un problème de conditionnement d'un récipient.

L'algorithme génétique vise à résoudre le problème suivant: l'emballage des boîtes rectangulaires avec plusieurs tailles différentes dans un récipient. Ce problème est connu sous le nom de bin-packing.

La nouveauté de cet algorithme génétique va introduire la solution de divers présentés à ce jour, est une nouvelle façon de créer des modèles initiaux, c'est-à HSSI est utilisé heuristique (Heuristique surfaces irrégulières lissage) qui vise à créer une population initiale de afin d'optimiser l'algorithme génétique.

L'heuristique HSSI essaie de résoudre les problèmes d'emballage simulant le comportement de la plupart des gens à ce processus dans la vraie vie, cependant, a un champ de recherche entre les petites et les solutions possibles seront ensuite utilisé un algorithme génétique pour étendre le champ de recherche et explorer de nouvelles solutions.

À la fin, il est destiné à obtenir d'un logiciel d'où il est possible de configurer un problème donné d'un récipient d'emballage et d'obtenir la solution de la même par l'algorithme génétique.

Ainsi, l'objectif principal de l'étude est de contribuer à la recherche et conclusions sur cette question et d'apporter une nouvelle proposition pour une solution au problème des emballages.

Mots-clés

Lot Emballage problème, heuristiques, algorithmes génétiques.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

API – Application Programming Interface

PCE – Problema de Corte e Empacotamento

AG – Algoritmo Genético

JGAP – Java Genetic Algorithms Package

CPU – Communications Processor Unit

1. INTRODUÇÃO

O problema de empacotamento de contentores é clássico na investigação operacional, geralmente o objetivo deste é determinar uma configuração de carga tal forma que o volume utilizado em relação ao volume total disponível pelo contentor seja maximizado, por outras palavras o principal objetivo é ocupar ao máximo o volume do contentor.

Na literatura este problema é distinguido em quatro variantes [Pisinger, 2002], uma em que toda a carga tem que ser armazenada, sendo possível usar mais do que um contentor com as mesmas dimensões este problema é conhecido como *Bin-Packing*, outra variante onde também é necessário carregar toda a carga em um ou em vários contentores mas com dimensões diferentes, este problema é conhecido como *Multi-container loading,* uma outra variante tolera que alguns itens sejam deixados para trás, mas utiliza apenas um contentor, este problema é conhecido como *Knapsack*. Na última variante é necessário empacotar toda a carga num contentor que tem duas dimensões fixas e a outra dimensão é variável, como por exemplo o empacotamento de uma palete. Este problema é conhecido como *Strip packing*.

Existem outras características que podem servir como diferenciação do problema, como por exemplo o tipo de carga que se pretende alocar ao contentor, que pode ser homogênea quando existe apenas um tipo de caixa, ou heterogênea quando existe vários tipos de caixa.

O problema também pode ser diferenciado através da liberdade de posicionamento dos objetos a carregar, ou seja, se estes objetos podem sofrer uma rotação em uma, duas ou em três dimensões ou se exige que alguma dimensão seja fixada, como por exemplo a altura.

Outras características podem-se prender com o objetivo do problema, ou seja, se este pretende maximizar o peso da carga, o equilíbrio (centro de gravidade) da carga, o valor monetário associado à carga, ou se o objetivo é minimizar a diferença entre os pesos da parte frontal e traseira, esquerda e direita do suposto contentor proporcionando a sua estabilidade. A classificação deste tipo de problema está diretamente relacionada com as suas restrições. Todos os esforços são usados para que os itens de carga sejam alocados ao contentor da melhor maneira possível, visando que os objetivos do problema sejam cumpridos.

O problema que será tratado neste trabalho é do tipo *Bin-Packing* com cargas fortemente homogêneas ou fortemente heterogêneas.

Este problema é considerado NP-difícil e portanto complexo de ser resolvido matematicamente e deterministicamente.

Por não se tratar de um problema simples, devido às suas restrições como peso, dimensões, centro de gravidade, orientação do posicionamento, valores monetários entre outras, é extremamente difícil a determinação de uma solução ótima. A solução do problema é extremamente complicada de ser encontrada analiticamente e em termos computacionais, é inviável nos dias de hoje tentar encontrar soluções deste problema através de uma abordagem matemática e determinística.

Neste tipo de problema justifica-se então a utilização de técnicas heurísticas ou metaheurísticas para uma resolução aproximada, levando à determinação de uma solução ótima ou próxima de ótima, mas interessante. O trabalho aqui apresentado faz uma pesquisa sobre as metaheurísticas que foram propostas para a resolução do problema de empacotamento de contentor e propõe um algoritmo genético como uma nova solução.

Este algoritmo genético vai solucionar um problema onde existe, um contentor a carregar e um número de caixas, com dimensões diferentes ou iguais entre si.

A singularidade deste algoritmo em relação aos algoritmos que existem na literatura deve-se, ao facto de utilizar uma determinada heurística, para gerar uma população inicial. A

razão que leva à necessidade de adotar esta estratégia prende-se, com a ineficiente do algoritmo genético quando trabalha com operadores genéticos aleatórios. A utilização de heurísticas para gerar uma população inicial, já foi ensaiada noutros algoritmos e verificou-se que, melhorou o desempenho do algoritmo.

A heurística que é utilizada para controlar a população inicial é a HSSI, esta baseia-se nos mecanismos do comportamento da maioria das pessoas, quando na vida real, tentam resolver problemas de empacotamento.

No dia-a-dia as pessoas revelam que ao tentarem solucionar um problema de empacotamento, onde existe vários objetos diferentes, tem a tendência de escolher sempre os objetos cujas dimensões se aproximem ao máximo do espaço disponível para o empacotamento. No caso de ainda existir espaço desocupado, a próxima caixa a ser escolhida é aquela que melhor preenche o espaço de forma a garantir que o topo de ambas as caixas forme uma superfície sem descontinuidade, ou seja, "suavizada". Este mecanismo de empacotamento faz com que no final se obtenha varias camadas de objetos, eventualmente camadas por cima de camadas ou camadas ao lado de camadas.

Contudo existe um problema ao recorrer-se a uma heurística para criar uma população inicial para o algoritmo genético, este problema reside no facto de a heurística explorar uma pequena parte do campo de soluções. Este método de pesquisa pode ignorar soluções mais interessantes que aquela que explorou. Para contornar esta limitação da heurística é introduzido um fator de aleatoriedade na HSSI, para apresentar uma população inicial diversificada e viável para o algoritmo genético utilizar, evitando assim soluções iniciais repetidas e inviáveis do ponto de vista das restrições.

Quando criada a população inicial o algoritmo genético vai então, recorrer ao operador de mutação para melhorar a solução inicial caso seja possível e fazer uma avaliação da aptidão da mesma, os operadores de seleção e cruzamento não vão interferir no processo de evolução solução, ou seja, vão estar "desativados".

Para criar este algoritmo vai ser utilizada a API JGAP (Java Genetic Algorithm Package) que é uma API escrita em Java disponibilizada gratuitamente e que fornece os mecanismos básicos dos algoritmos genéticos. Estes mecanismos são: gerador da população inicial, operadores de evolução e avaliação da solução, sendo necessário configurar cada um deles.

No final é esperado obter um *software* para ser utilizado em qualquer problema de empacotamento onde existe um contentor e varias caixas a empacotar. Este *software* vai ser construído em Java e através dos componentes gráficos que a API Swing disponibiliza.

1.1. Contextualização

Este trabalho surgiu como resultado da dissertação do Mestrado em Engenharia Eletrotécnica e de Computadores no ramo de Sistemas e Planeamento Industrial. O interesse pelo tema de empacotamento de contentores ocorreu por se verificar que hoje em dia é cada vez mais útil e necessário, que as empresas se dotem de técnicas que as ajudem a tomar decisões mais racionais e eficientes.

O estudo desenvolvido neste trabalho está direcionado à elaboração de uma ferramenta de auxílio à tomada de decisão, a nível da distribuição e transporte para as empresas, ou em relação a problemas pessoais do quotidiano.

1.2. Objetivos

O objetivo principal deste trabalho é fazer uma pesquisa sobre o problema de empacotamento de contentores e criar uma ferramenta para solucionar problemas de empacotamento, onde existe um contentor e vários objetos regulares a empacotar.

Processo de desenvolvimento:

1. Fazer um estudo sobre o estado de desenvolvimento do problema de empacotamento de contentores;
2. Criar um algoritmo genético para resolver problema de empacotamento;
3. Criar um *software* e implementar o algoritmo genético;

1.3. Calendarização

Este trabalho é desenvolvido num período de 5 meses e é distribuído em 8 tarefas como mostra a Figura 1.

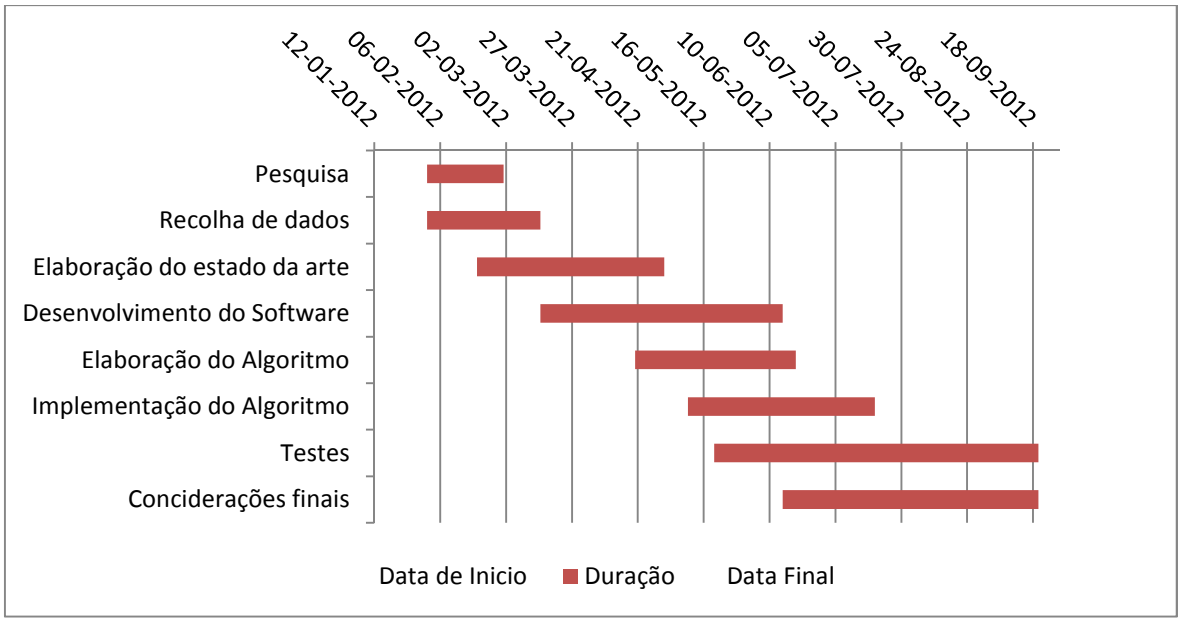


Figura 1 Calendarização do trabalho

1.4. Organização do relatório

Este relatório está dividido em 6 capítulos, no capítulo 1 é feita uma introdução ao tema e são explicados os objetivos do trabalho. No capítulo seguinte, capitulo 2, consta uma apresentação sobre o estado do problema de empacotamento em relação às abordagens e propostas que foram aparecendo até aos dias de hoje. O capítulo 3 expõe o estado da arte dos algoritmos genéticos, como surgiram e os desenvolvimentos que sofreram. No capítulo 4 são apresentadas as metaheurísticas que foram propostas para resolver este problema. O capítulo 5 é dedicado ao desenvolvimento de um algoritmo genético que solucione este problema, neste capítulo estão expostas as fases de desenvolvimento, desde a primeira versão do algoritmo até à final e conclusões de cada versão. No último capítulo, são reunidas as considerações finais e perspetivas de melhoramento no futuro.

2. PROBLEMA DE EMPACOTAMENTO

Este capítulo é dedicado à apresentação do problema de empacotamento de contentores, a origem do problema, uma possível formulação matemática do problema, as primeiras soluções que surgiram e ainda as propostas de solução mais relevantes que foram exibidas até a data.

Procura-se com este capítulo fazer um enquadramento do estado atual do problema de empacotamento e mostrar a complexidade do problema, para assim justificar a aplicação de um método heurístico ou metaheurístico, como é o caso do algoritmo genético, para a resolução deste problema.

2.1. Introdução

O problema de empacotamento de contentores (PEC) é muito conhecido na área da investigação operacional, geralmente o objetivo deste é determinar uma configuração ou

disposição da carga, de tal modo que o volume utilizado pela carga em relação ao volume total disponibilizado pelo contentor seja maximizado, ou seja, aproveitar ao máximo o volume do contentor.

Dyckhoff em 1990 destacou que as características e a estrutura básica do problema de corte ou de empacotamento são idênticas, sugere também que deve haver uma anexação destas duas áreas de pesquisa que até então eram tratadas isoladamente [Dyckhoff, 1990]. O problema passou assim ter uma designação mais geral Problemas de Corte e Empacotamento (PCE).

Os PCE foram inicialmente estudado no trabalho pioneiro do matemático e economista soviético Leonid Vitaliyevich Kantorovich publicado em 1939, onde o autor apresenta vários modelos matemáticos de programação linear para problemas de planeamento e organização da produção, [Leonid, 1960].

Entre os vários problemas destacados neste trabalho, está o Problema de Corte e Empacotamento Unidimensional, mas como consequência da 2ª Guerra Mundial e da Guerra Fria este trabalho não foi divulgado ao Ocidente até 1959, sendo depois publicado em inglês em 1960.

Na década de sessenta surgiu então pela primeira vez uma solução viável para o problema de corte e empacotamento unidimensional, através de um método heurístico apresentado em [Gilmore e Gomory, 1961 e 1965] e desde então foram apresentados vários documentos que sugerem formas alternativas de solução.

Nas últimas décadas, os PCE ganharam uma atenção especial de muitos investigadores interessados devido as diversas aplicações do problema ou à dificuldade relacionada à sua resolução uma vez que a maioria dos PCE pertence á classe NP-difíceis [Garey e Johnson, 1979], sendo estes não resolúveis por um algoritmo polinomial, a menos que a classe P = NP, tornando o algoritmo instável para obtenção da solução por métodos exatos para problemas de grande porte. Esta relevância dos PCE é despertada em parte pela necessidade das indústrias se tornarem mais competitivas e melhorarem o custo/benefício dos seus produtos. Esta necessidade torna imprescindível a racionalização de recursos, procurando assim soluções que minimizem as perdas, como os problemas de corte, ou maximizem a utilização de espaços, como os problemas de empacotamento.

Os PCE podem ser encontrados em vários tipos de indústrias, como por exemplo a indústria metalúrgica, da madeira, papel, vidro, móveis, plástico, entre outras e nos serviços de logística. Um dos reflexos da importância destes problemas é o ESICUP (EURO *Special Interest Group on CUtting and Packing Problems*), que é uma organização internacional cujo objetivo é reunir e difundir os trabalhos na área.

2.2. Características do Problema

Os Problemas de Corte e Empacotamento podem ser classificados em relação às dimensões que são relevantes durante o processo de corte ou empacotamento. Na tipologia de Dyckhoff, [Dyckhoff, 1990], esta caracterização é feita devido ao papel fundamental desempenhado pelos padrões de corte e da natureza de combinações geométricas (combinar itens sobre objetos) podendo ter dimensões espaciais e não-espaciais.

No caso dos problemas com dimensões espaciais, os objetos e itens podem ter uma, duas ou as três dimensões do espaço euclidiano relevantes para o processo de corte ou empacotamento no caso não-espaciais podem ter dimensões como tempo, peso, memória, etc.

2.2.1. Unidimensional

Um PCE é dito unidimensional quando um objeto (bobine, barra, etc.) deve ser cortado ao longo do seu comprimento para criar itens de comprimentos menores pré-determinados. Cada item tem um "valor de utilidade" associado. Itens cujos comprimentos não foram especificados são considerados perdas, com valor de utilidade nulo. Este é um problema de otimização combinatória, que busca obter os itens de tamanhos especificados a partir do corte de objetos de tal forma que se obtenha o maior valor de utilidade total, dado pela soma dos valores de utilidade dos itens produzidos. A Figura 2 mostra o corte de uma bobina grande em bobinas menores.

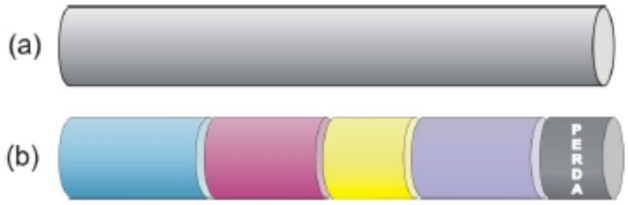


Figura 2 (a) Barra; (b) barra cortado em 4 itens.

2.2.2. Bidimensional

O PCE Bidimensional tal como o unidimensional pretende obter um conjunto de itens menores, com várias dimensões pré-definidas, que vão resultar do corte de objetos maiores, ou alocar objetos em espaços com dimensões determinadas. Nestes casos, o comprimento e a largura são restrições do processo de corte ou empacotamento. A Figura 3 mostra o corte de um objeto retangular.

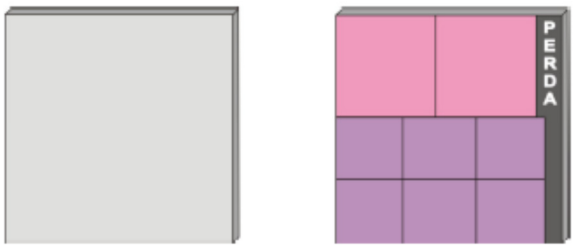


Figura 3 (a) Placa; (b) placa cortada em 8 itens.

2.2.3. Tridimensional

Um PCE é dito tridimensional quando as três dimensões do objeto (largura, comprimento e altura) são relevantes durante o processo de corte ou empacotamento. Este caso têm uma grande aplicabilidade na prática em problemas de empacotamento e tem como principal objetivo minimizar os espaços desocupados dentro do espaço a armazenar. A Figura 4 ilustra a alocação de itens em três dimensões. Este exemplo pode ser visto como um empacotamento de um contentor.

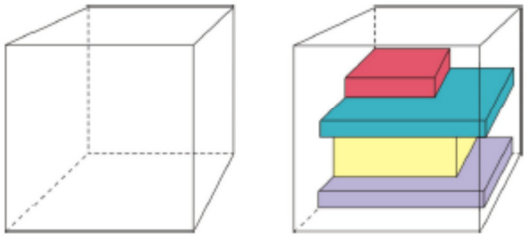


Figura 4 (a) Contentor; (b) 4 caixas empacotadas no contentor

2.2.4. 1.5-dimensional

Acontece quando duas dimensões são relevantes para a solução do problema. Porém, apenas uma delas tem um tamanho fixo. Um exemplo desse tipo de problema é o de um rolo de tecido que tem largura fixa e comprimento suficientemente para a produção de roupas. A Figura 5 ilustra o problema de corte 1.5 - dimensional, em que o comprimento total cortado L deve ser minimizado.

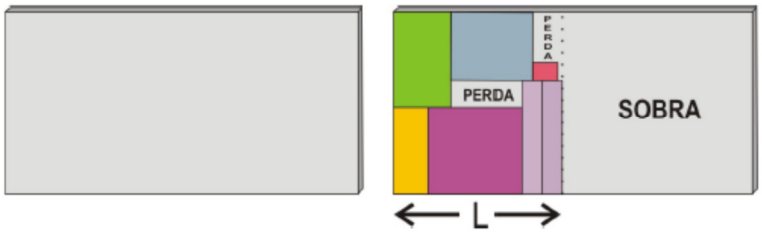


Figura 5 (a) Rolo; (b) rolo cortado em itens.

2.2.5. 2.5-dimensional

Três dimensões são consideradas para a resolução do problema, porém uma delas é variável. Um exemplo é o empacotamento de uma palete com largura e comprimento fixo e com altura suficientemente para acomodar um volume de carga. A Figura 5 pode ser interpretada como uma visão lateral deste caso.

2.2.6. N-dimensional

Estes problemas fazem parte dos problemas de dimensões não-espaciais na associação de Dyckhoff, [Dyckhoff, 1990], podem ser considerados problemas abstratos, pois as

dimensões levadas em consideração não são dimensões físicas ou palpáveis. Exemplos deste tipo de problema são os problemas de alocação de tarefas, a programação de rotas de veículos, navios e aviões, a alocação de memória de computador, entre outros. Nestes problemas, os objetos e os itens têm uma métrica diferente, isto é, ao invés de terem comprimento, largura e altura, podem ter dimensões como tempo, peso ou memória, que são não-espaciais.

2.3. Tipologias

Devido à grande diversidade de PCE que vieram a aparecer desde os anos sessenta, alguns autores desenvolveram critérios para caracterizá-los a fim de simplificar o desenvolvimento ou a escolha de métodos de solução para cada caso.

Assim, Dyckhoff, [Dyckhoff, 1990], expôs uma tipologia para qualificar conforme algumas particularidades. A intenção desta tipologia foi de agregar os diferentes usos de PCE na literatura e concentrar pesquisas futuras sobre tipos de problemas melhor definidos. As 4 características principais da tipologia são e estão subdivididas nos seguintes tipos:

(i) **Dimensão do problema**

(1) unidimensional

(2) bidimensional

(3) tridimensional

(N) N- dimensional (N>3)

(ii) **Forma de alocação das unidades**

(V) seleção de unidades grandes

(B) seleção de unidades pequenas

(iii)**Variedade de unidades grandes**

(O) uma unidade

(I) unidades de tamanhos iguais

(D) unidades de tamanhos diferentes

(iv) **Variedade de unidades pequenas**

(F) poucas unidades de tamanhos diferentes

(M) muitas unidades de muitos tamanhos diferentes

(R) muitas unidades de poucos tamanhos diferentes

(C) unidades de tamanhos iguais

Combinando todos os tipos destas 4 características principais, obtém-se 96 tipos diferentes para os problemas de corte e empacotamento.

A falta de uma característica na combinação das características indica que o PCE pode ocorrer em várias situações definidas. O problema clássico da mochila ou *Knapsack* é representado por 1/B/O/ enquanto o problema do empacotamento de contentores com caixas idênticas é representado pela notação 3/B/O/C.

Embora tenha destacado a estrutura comum entre os PCE, a tipologia de Dyckhoff, [Dyckhoff, 1990], não obteve unanimidade no meio científico e Wäscher em [Wäscher *et al.,* 2007] destaca as desvantagens da tipologia de Dyckhoff, [Dyckhoff, 1990].

As principais desvantagens são as de não atribuir a cada problema uma única codificação e de não se obter categorias de problemas homogêneos.

Sendo assim em 2007 Wäscher, [Wäscher *et al.,* 2007], propõem que a definição dos tipos de PCE utilize os seguintes cinco critérios:

(i) **Dimensionalidade**

É distinguida em 1, 2 ou 3 dimensões, similarmente à tipologia de Dyckhoff, [Dyckhoff, 1990].Problemas com mais de 3 dimensões são considerados variantes dos anteriores.

(ii) **Tipo de designação**

Corresponde ao critério "Forma de alocação das unidades" de Dyckhoff, [Dyckhoff, 1990], e considera duas situações básicas:

- Problemas do tipo maximização de saídas (*output maximisation types*): um dado conjunto de itens deve ser designado a um dado conjunto de objetos e o conjunto de objetos não é suficiente para acomodar todos os itens. Neste caso, todos os objetos serão utilizados e será selecionado o subconjunto de itens de máximo valor total.

- Problemas do tipo minimização de entradas (*input minimisation types*): um dado conjunto de itens cuja procura deve ser satisfeita completamente deve ser designado a um dado conjunto de objetos em número suficiente para acomodar todos os itens. Desta forma, todos os itens serão designados e será selecionado o subconjunto de objetos de mínimo valor total.

(iii) Variedade de itens

Corresponde ao critério "Variedade de unidades pequenas" de Dyckhoff, [Dyckhoff, 1990], sendo os problemas classificados como tendo:

(i) Itens idênticos
(ii) Variedade fracamente heterogênea
(iii) Variedade fortemente heterogênea

A classificação (*i)* corresponde à situação em que os itens possuem a mesma forma e tamanho e é idêntica à categoria C de Dyckhoff, [Dyckhoff, 1990]. A classificação (*ii)* corresponde aos casos em que existem muitos itens de poucos tamanhos/formas diferentes, e é idêntica à categoria R de Dyckhoff, [Dyckhoff, 1990]. Finalmente, na classificação (*iii)* existe quando os itens têm muitos tamanhos/formas diferentes, correspondendo às categorias F e M de Dyckhoff, [Dyckhoff, 1990].

(iv) Variedade de objetos

Corresponde ao critério "Variedade de unidades grandes" de Dyckhoff, [Dyckhoff, 1990], e considera dois tipos principais: (*i*) um objeto e (*ii*) vários objetos. No caso de um único objeto (tipo O de Dyckhoff, [Dyckhoff, 1990]), admite-se que o mesmo possui todas as dimensões fixas ou uma ou mais dimensões variáveis. No caso de vários objetos, admite-se que os mesmos são idênticos (tipo I de Dyckhoff, [Dyckhoff, 1990]), tem variedade fracamente heterogênea, ou variedade fortemente heterogênea. Para a definição da tipificação básica de problemas, admite-se que para casos de duas e três dimensões, todos os objetos têm forma retangular e consistem de material homogêneo. Objetos não-retangulares e/ou não homogêneos resultam em variantes dos problemas anteriores.

(v) Forma dos itens

Em relação a problemas de 2 e 3 dimensões, os itens são distinguidos entre regulares (retângulos, círculos, caixas, cilindros, bolas, etc.) e irregulares. De acordo com o que é geralmente assumido na literatura, itens retangulares são posicionados ortogonalmente nos objetos, ou seja, os itens são dispostos paralelamente às faces do(s) objeto(s). Problemas que permitem *layouts* não-ortogonais e/ou mistos de itens regulares e irregulares são vistos como variantes do problema de itens retangulares posicionados ortogonalmente.

Com base nas características acima, Wäscher, [Wäscher *et al.,* 2007], propõem para os PCE, as tipificações básica, intermediária e refinada, descritas a seguir.

A. Tipos básicos

Os tipos básicos contemplam as categorias "Problemas do tipo maximização de saídas" e "Problemas do tipo minimização de entradas", levando em consideração a combinação dos critérios "Tipo de designação" e "Variedade de itens". A Figura 6 mostra as combinações relevantes e problemas associados.

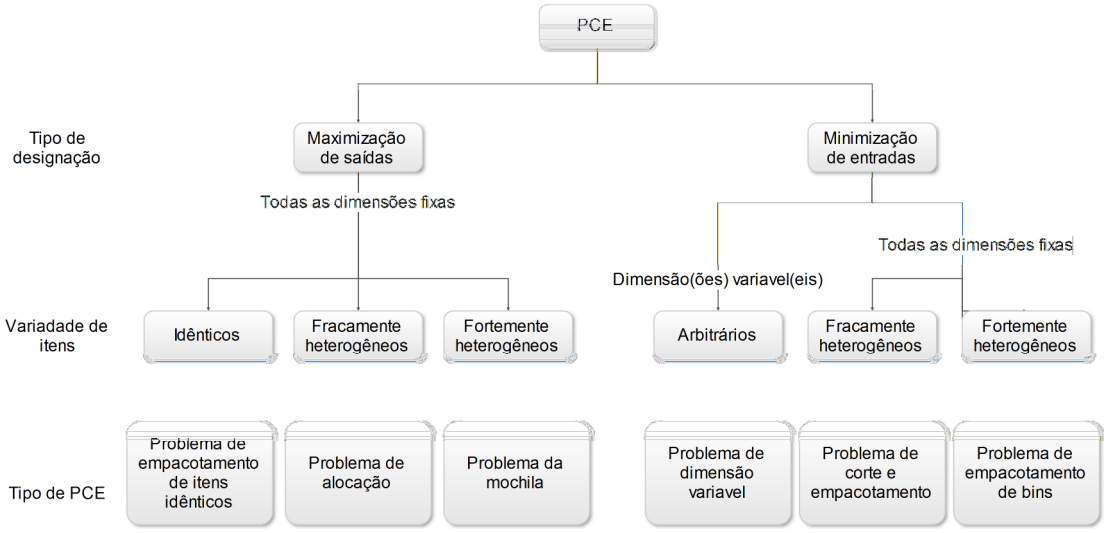


Figura 6 Tipos básicos.

B. Tipos intermédios

Para definir os tipos intermédios é adicionado o critério "Variedade de objetos" aos problemas utilizados para os tipos básicos. Desta forma, obtêm-se uma maior diferenciação entre os problemas. As Tabelas 1 e 2 sumarizam o sistema de problemas do tipo minimização de entradas e maximização de saídas, respetivamente. As siglas utilizadas nas tabelas são abreviaturas do termo em inglês de cada tipo do problema.

Tabela 1 Tipos intermédios/minimização de entradas.

			Variedade de itens	
			Fracamente heterogêneos	Fortemente heterogéneo
Características dos objetos	Todas as dimensões fixas	Idênticos	Problema de corte e de empacotamento de objetos de um único tamanho SSSCSP	Problema de empacotamento de *bins* de um único tamanho SBSBPP
		Fracamente heterogêneos	Problema de corte e de empacotamento de objetos de tamanhos diferentes MSSCSP	Problema de empacotamento de *bins* de tamanhos diferentes SBSBPP
		Fortemente heterogéneo	Problema de corte e de empacotamento residual RCSP	Problema de empacotamento de *bins* residual RBPP

	Apenas um objeto com uma ou mais dimensões variáveis	Problema de dimensão "aberta" ODP

Tabela 2 Tipos intermédios/maximização de saídas.

			Variedade de itens		
			Idênticos	Fracamente heterogêneos	Fortemente heterogéneo
Características dos objetos	Todas as dimensões fixas	Um objeto	Problema de empacotamento de itens idênticos IIPP	Problema de alocação em um único objeto SLOPP	Problema da mochila SKP
		Idênticos		Problema de alocação em múltiplos objetos idênticos MILOPP	Problema de múltiplas mochilas idênticas MIKP
		Heterogéneos		Problema de alocação em múltiplos objetos heterogêneos MHLOPP	Problema de múltiplas mochilas heterogêneas MHKP

C. Tipos refinados

A obtenção de tipos refinados é feita adicionando o critério "Dimensionalidade" aos tipos intermédios para problemas com duas dimensões e o critério "Forma dos itens" para três dimensões. As subcategorias resultantes são caracterizadas por adjetivos adicionados aos nomes dos problemas do tipo intermédio de acordo com o seguinte sistema: {dimensão} + {forma} + {IPT}

Em que:

- Dimensão: 1,2 ou 3;
- Forma: retangular, circular, irregular,…;
- IPT: nome do problema do tipo intermédio.

Wäscher em [Wäscher *et al.,* 2007], afirma que esta tipologia fornece uma estruturação mais abrangente e precisa dos PCE do que a proposta em [Dyckhoff, 1990], e permite constatar a existência de uma grande diversidade de tipos de problemas.

2.3.1. Variantes do Problema de Empacotamento

Na literatura técnica, existem várias definições sobre o PEC. Uma das definições encontradas sobre este problema é descrita em [Bortfeldt e Gehring, 2001] onde o problema é apresentado como um conjunto de pacotes retangulares (caixas) que devem ser arranjados em um ou mais contentores retangulares de tal maneira que o melhor uso do espaço disponível do contentor seja feito para o armazenamento da carga. O empacotamento ótimo de um contentor reduz o custo de transporte como também aumenta a estabilidade e apoio da carga.

Há, porém, diversas variantes do PEC que dependem da maneira como o padrão de empacotamento é avaliado e/ou restrições que apresentam.

Segundo Bischoff e Marriott, [Bischoff e Marriott, 1990], os problemas de empacotamento de contentores aparecem na prática em dois casos importantes:

(i) Quando uma combinação de diferentes contentores deve ser escolhida para transportar uma dada carga ($m \geq 1$ e $n \geq 1$). Este problema pode ser classificado em Dyckhoff, [Dyckhoff, 1990], como 3/V/ / em que os últimos 2 valores omitidos correspondem respetivamente à variedade de unidades grandes (objetos) e à variedade de unidades pequenas (itens). [Wäscher *et al.,*2007] incluem estes problemas dentre as classes SSSCSP, SBSBPP, MSSCSP e MBSBPP (ver Tabela 2).

(ii) Quando o maior volume de uma dada carga deve ser escolhido para ser transportado num único contentor. Em relação à variedade de caixas, o empacotamento é classificado como homogêneo quando as caixas são idênticas, e heterogêneo, caso contrário. Neste último caso, o empacotamento pode ser ainda classificado como fortemente heterogêneo caso haja muitos tipos de caixas, e como fracamente heterogêneo se o número de tipos de caixas for

pequeno. De acordo com a classificação de Dyckhoff, [Dyckhoff, 1990], quando as caixas são idêntico, o problema é classificado como 3/B/O/C; se a carga é fortemente heterogênea, o problema é classificado como 3/B/O/M. Em [Wäscher *et al.,* 2007], esses 2 problemas podem ser classificados respetivamente como IIPP e SKP (veja Tabela.2).

Além de restrições geométricas (a carga deve caber dentro do contentor e as caixas não podem estar sobrepostas), podem-se considerar várias outras, desta vez dependendo do tipo de produto a ser transportado, ou mesmo impostas pela empresa responsável pela armazenagem e distribuição, como limitações de custos. Pisinger, [Pisinger, 2002], apresentou as seguintes variantes do PEC de acordo com a função objetivo e restrições presentes:

2.3.2. Empacotamento por faixas (*Strip packing*).

O contentor tem a largura e altura fixa, e o problema consiste em empacotar todas as caixas de forma que a profundidade utilizada do contentor seja mínima. Este problema é aplicado em situações *multi-drop*, onde o empacotamento deve ser dividido em áreas distintas do contentor, cada qual correspondendo a um determinado destino.

2.3.3. Empacotamento da Mochila (*Knapsack loading*)

Cada caixa está associada a um valor de lucro, e o problema consiste em escolher um subconjunto de caixas que maximizem o lucro. Se a lucratividade de cada caixa está associada ao volume, então o objetivo corresponde a maximizar o espaço utilizado do contentor.

2.3.4. Bin-packing

Um número limitado de caixas deve ser empacotado utilizando o menor número de contentores idênticos.

2.3.5. Empacotamento de múltiplos contentores (Multi-container *loading*)

Similar ao *Bin-Packing*, exceto que os contentores podem ter diferentes dimensões e o objetivo é o de escolher um subconjunto de contentores para o qual os custos de transporte sejam minimizados.

2.4. Formulação do Problema

O modelo aqui apresentado tem como base o modelo apresentado em 1995 por Chen, Lee, [Chen, Lee, et al., 1995], onde desenvolveram um modelo analítico para o problema de empacotamento de contentores, utilizando variáveis inteiras e binárias, onde são consideradas caixas retangulares e de tamanhos não uniformes. O modelo prevê também a possibilidade de um empacotamento em diferentes contentores de dimensões possivelmente diferentes entre si. No seu modelo matemático, o objetivo é encontrar uma solução que minimize o espaço desperdiçado do contentor e minimize o número de contentores necessários para carregar todas as caixas.

No modelo aqui apresentado, supõe-se que existe apenas um contentor para carregar e que as dimensões das caixas e do contentor são conhecidas, bem como a quantidade de caixas a serem carregadas. As caixas são independentes e são posicionadas ortogonalmente no contentor, ou seja, são posicionadas em paralelo ou perpendicular aos eixos como indica a Figura 7 que faz uma descrição do enquadramento deste modelo.

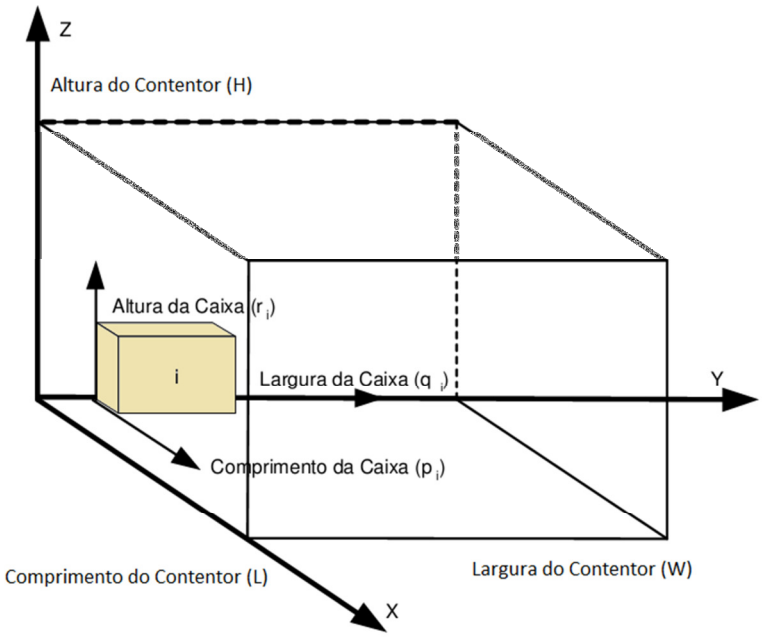


Figura 7 Ilustração do posicionamento da caixa, de acordo com suas dimensões e as dimensões do contentor.

As caixas podem sofrer rotações no seu posicionamento, de tal forma que a largura, o comprimento e a altura da caixa não sejam posicionados obrigatoriamente em paralelo á largura, ó comprimento e á altura do contentor respetivo.

As notações listadas abaixo são necessárias para o entendimento do modelo matemático:

$\boldsymbol{N}$ – número de caixas disponíveis para empacotamento;

$\boldsymbol{si}$ - variável binária que indica se a caixa foi posicionada no contentor. Quando isso ocorre, $si = 1$, caso contrário, si = 0;

$(\boldsymbol{L}; \boldsymbol{W}; \boldsymbol{H})$ - indica o comprimento, largura e altura do contentor, respetiva;

$(\boldsymbol{pi};\ \boldsymbol{qi};\ \boldsymbol{ri})$ - indica o comprimento, a largura e altura da caixa, respetiva;

$(\boldsymbol{xi};\ \boldsymbol{yi};\ \boldsymbol{zi})$ - indica a localização da caixa pelo canto inferior esquerdo traseiro (vértice mínimo);

$(\boldsymbol{lxi};\ \boldsymbol{lyi};\ \boldsymbol{lzi})$ - variável binária que indica para qual eixo o comprimento da caixa está em paralelo.

$(\boldsymbol{wxi};\ \boldsymbol{wyi};\ \boldsymbol{wzi})$ - variável binária que indica para qual eixo a largura da caixa está em paralelo.

$(\boldsymbol{hxi};\ \boldsymbol{hyi};\ \boldsymbol{hzi})$ - variável binária que indica para qual eixo a altura da caixa está em paralelo.

Ainda existem outras variáveis que são usadas para indicar o posicionamento das caixas em relação a outras caixas:

aik - caso seja 1, indica que a caixa i está à esquerda da caixa k;

bik - caso seja 1, indica que a caixa i está à direita da caixa k;

cik - caso seja 1, indica que a caixa i está atrás da caixa k;

dik - caso seja 1, indica que a caixa i está à frente da caixa k

eik - caso seja 1, indica que a caixa i está abaixo da caixa k;

fik - caso seja 1, indica que a caixa i está acima da caixa k;

O problema então é formulado conforme abaixo, onde se observa que o objetivo é

Minimizar o espaço não preenchido do contentor:

$$min\ L.W.H - \sum_{i=1}^{N} pi.qi.ri.si$$

Sujeito a:

(Evitar sobreposições de caixas no contentor)

$$x_i + p_i.l_{xi} + q_i.w_{xi} + r_i.h_{xi} \leq x_k + (1 - c_{ik}).M, \forall_i, k, i < k \qquad (1)$$

$$x_k + p_k.l_{xk} + q_k.w_{xk} + r_k.h_{xk} \leq x_i + (1 - d_{ik}).M, \forall_i, k, i < k \qquad (2)$$

$$y_i + p_i.l_{yi} + q_i.w_{yi} + r_i.h_{yk} \leq y_k + (1 - a_{ik}).M, \forall_i, k, i < k \qquad (3)$$

$$y_k + p_k.l_{yk} + q_k.w_{yk} + r_k.h_{yk} \leq y_i + (1 - b_{ik}).M, \forall_i, k, i < k \qquad (4)$$

$$z_i + p_i.l_{zi} + q_i.w_{zi} + r_i.h_{zi} \leq z_k + (1 - e_{ik}).M, \forall_i, k, i < k \qquad (5)$$

$$z_k + p_k.l_{zk} + q_k.w_{zk} + r_k.h_{zk} \leq y_i + (1 - b_{ik}).M, \forall_i, k, i < k \qquad (6)$$

(Garantir que o par de caixas avaliados nas equações (1) a (6) está no contentor)

$$a_{ik} + b_{ik} + c_{ik} + d_{ik}. + e_{ik} + f_{ik} \geq s_i + s_k - 1, \forall_i, k, i < k \qquad (7)$$

(Garantir que o posicionamento das caixas obedeça as limitações físicas dimensionais do contentor)

$$x_i + p_i.l_{xi} + q_i.w_{xi} + r_i.h_{xi} \leq L + (1 - s_i).M, \forall_i \quad (8)$$

$$y_i + p_i.l_{yi} + q_i.w_{yi} + r_i.h_{xi} \leq W + (1 - s_i).M, \forall_i \quad (9)$$

$$z_i + p_i.l_{zi} + q_i.w_{zi} + r_i.h_{zi} \leq H + (1 - s_i).M, \forall_i \quad (10)$$

M é um número inteiro arbitrário e grande.

As variáveis binárias que identificam o posicionamento das caixas com relação aos eixos são variáveis dependentes e se relacionam conforme abaixo:

$$l_{xi} + l_{yi} + l_{zi} \leq 1 \quad (11)$$

$$w_{xi} + w_{yi} + w_{zi} \leq 1 \quad (12)$$

$$z_{xi} + z_{yi} + z_{zi} \leq 1 \quad (13)$$

$$l_{xi} + w_{yi} + z_{zi} \leq 1 \quad (14)$$

$$l_{yi} + w_{yi} + z_{yi} \leq 1 \quad (15)$$

$$l_{zi} + w_{zi} + z_{zi} \leq 1 \quad (16)$$

Apesar de ter existido uma simplificação ao limitar o numero de contentores a um o problema continua a ser complexo e a ser classificado como um problema de classe NP-Complete (NP completo) (Non-deterministic Polynomial-time Complete), o que significa que o problema não apresenta uma solução polinomial.

2.5. Técnicas e Soluções

Tem surgido várias propostas para a resolução de problemas NP-Completo ou problemas NP-Difícil e verifica-se que os métodos aproximados têm sido cada vez mais utilizados nestes problemas.

As causas desta tendência estão no facto de os métodos exatos não conseguirem determinar uma solução, de problemas complexos num tempo interessante, como o problema de empacotamento de contentores.

Entre os métodos exatos destaca-se o algoritmo *Brach-and-Bound* que se baseia na técnica de dividir os problemas em vários problemas de tamanho menor.

Nos métodos aproximados existem várias abordagens como as heurísticas, as metaheurísticas ou as técnicas especiais. Entre as heurísticas encontra-se a heurística de construção, gulosa e partição. Estas heurísticas são muito úteis em determinados problemas mas apresentam a desvantagem de não serem abrangentes a qualquer problema em relação as metaheurísticas.

O processo de funcionamento de uma heurística é realizar um conjunto de transições através do espaço de busca do problema, o processo é iniciado num determinado ponto do espaço de busca e termina num ponto de local ótimo.

As metaheurísticas ostentam como base de funcionamento componentes probabilísticos, o que as torna capazes de resolver qualquer problema independentemente da natureza do problema a ser tratado, o que motiva o uso destas. Em sumula estas técnicas conseguem evitar locais ótimos, por realizarem a procura da solução ótima em diversas regiões do espaço, utilizando uma aleatoriedade "direcionada" pela heurística codificada na função de aptidão.

Algumas das técnicas classificadas como metaheurísticas desenvolvidas e utilizadas nos últimos anos são: Algoritmos Genéticos, Busca Tabu, Simulated Annealing, Grasp, VNSn entre outras.

Rodrigues, [Rodrigues, 2000], resume os métodos de resolução de problemas de otimização em dois grupos: métodos exatos e métodos aproximados (heurísticas, meta heurísticas e especiais), conforme indica a Figura 8.

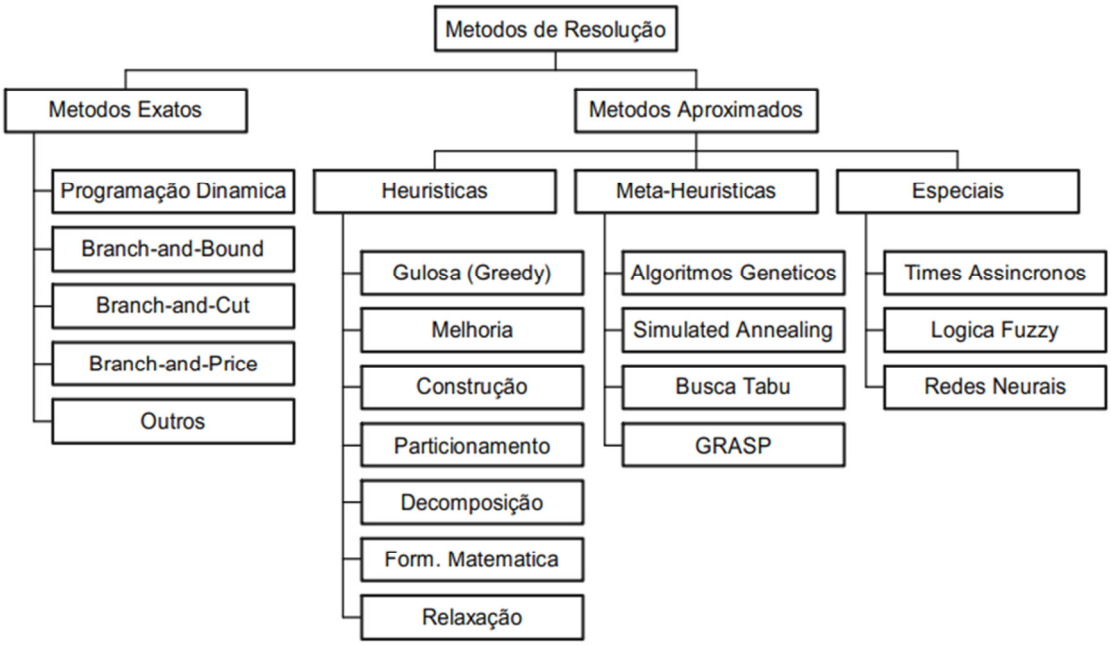


Figura 8 Métodos de Resolução de Problemas de Otimização [Rodrigues, 2000]

Os principais métodos de metas heurísticas citados acima podem ser resumidos conforme abaixo, no capítulo 4 será abordado de forma mais detalhada algumas das aplicações que existe deste método nos PEC:

- **Arrefecimento Simulado:** é uma técnica de busca local probabilística, que se fundamenta numa analogia com a termodinâmica, simulando o arrefecimento de um conjunto de átomos aquecidos. A técnica utiliza uma solução inicial aleatória para fazer uma busca pela solução final. A cada iteração é gerado aleatoriamente um vizinho da solução corrente e é verificado se esta corresponde á que se pretende encontrar.

- **Busca Tabu:** é um método de pesquisa que através de um procedimento adaptativo utiliza uma estrutura de memória como guia, que, por sua vez, armazena os últimos movimentos realizados, e é chamada de lista tabu. Essa lista funciona como uma fila, onde um novo movimento é adicionado e o mais antigo retirado e permite que se decida pela continuidade da exploração do espaço de soluções mesmo na

ausência de movimentos de melhora, evitando que haja retorno a um local ótimo previamente visitado.

- **Algoritmos Genéticos:** são algoritmos que se baseiam nos processos naturais de evolução, onde as características de uma possível solução são medidas por uma função de aptidão, determinando assim que os indivíduos mais aptos têm maiores probabilidades de sobreviver e reproduzirem-se. Para se reproduzirem ou criar uma nova geração, novamente em analogia com a biologia, operações de cruzamento, mutação e seleção são utilizadas. Quando uma população apresenta um determinado nível de evolução ou aptidão, o método termina e é selecionado o indivíduo (cromossoma) que apresenta as características que melhor satisfazem o problema.

- **GRASP:** (Greedy Randomized Adaptive Search Procedure) ou Procedimento de Busca Adaptativa Gulosa e Aleatória é um método interativo de duas etapas. A primeira etapa consiste na construção de soluções, elemento a elemento, e numa segunda etapa faz-se uma busca local para determinar um ótimo local na vizinhança de uma solução construída. O processo de seleção de soluções é baseado numa função adaptativa gulosa. Em cada iteração são geradas diferentes soluções.

2.6. Formas de Empacotamento

A forma como os itens são colocados é fundamental para a configuração final do empacotamento. O interesse em definir esta forma prende-se com o facto de otimizar o método de pesquisa que se pretende aplicar ao PEC. Existem várias formas de alocar os itens as mais conhecidas são:

- **Empacotamento em Parede**
- **Empacotamento em Camadas horizontais**
- **Empacotamento em Colunas**
- **Empacotamento em Blocos**

2.6.1. Empacotamento em Parede

Um dos objetivos desta forma de empacotamento é obter paredes de dimensão apropriada não muito estreitas nem muito largas. A largura de cada parede é circunscrevida pela dimensão da primeira caixa a ser colocada nessa parede. Assim a escolha da primeira caixa é fundamental que cumpre determinadas regras. Quando uma nova parede é criada que só é criada quando a parede anterior estiver preenchida, a escolha da primeira caixa determinará a largura da parede e assim sucessivamente.

No final da criação de uma parede, poderão existir espaços livres para os quais não existem caixas que aí possam ser colocadas apesar de o método de preenchimento desta heurística, levar a que estes espaços sejam contíguos à parede seguinte. Quando estes espaços vazios existem é possível unir estes espaços durante o empacotamento da próxima parede. Esta união de espaços livres em paredes adjacentes procura obter um empacotamento compacto e eficiente. Isto pode permitir o empacotamento de uma caixa de largura superior à da parede que está a ser empacotada. Apresentado assim, um empacotamento compacto, mas com intersecção entre as paredes.

2.6.2. Empacotamento em Camadas horizontais

Além do empacotamento em paredes, também é comum verificar heurísticas em que o empacotamento é construído através de combinações em camadas horizontais. Os trabalhos apresentado por [Scheithauer, 1991], [Loh e Nee, 1992] e [Bischoff et al., 1995] ilustram esta forma de empacotar.

Scheithauer, [Scheithauer , 1991], propõem um algoritmo para o problema Bin Packing. O empacotamento consiste na colocação de camadas horizontais ao longo da secção da base do contentor. Para empacotar as caixas em cada uma das camadas é utilizado um algoritmo de empacotamento bidimensional. O algoritmo ordena as caixas decrescentemente mediante a largura das mesmas e aloca-as em colunas horizontais por esta ordem.

Uma nova camada é construída quando a próxima caixa da lista não pode ser colocada na camada em construção. [Loh e Nee, 1992], propuseram também uma abordagem em camadas horizontais para resolução do PEC. As caixas têm dimensões verticais fixas e estão reunidas de acordo com essas dimensões. O objetivo é colocar caixas com a mesma

altura próximas umas das outras, cria-se então uma maior área plana para a posterior colocação de outras caixas.

2.6.3. Empacotamento em Colunas

Uma outra forma de fazer o empacotamento num contentor é através da colocação de caixas em colunas. Este tipo de empacotamento foi apresentado em [Bischoff e Ratcliff, 1995] e [Gehring e Bortfeldt, 1997] que adotaram este tipo de abordagem para resolução de um problema.

Em [Bischoff e Ratcliff, 1995], apresentam uma abordagem de empacotamento em colunas para o problema de empacotamento num só contentor. Este método proposto tem muitas características semelhantes com as abordagens de alocação em paredes. Em cada iteração do empacotamento, é aplicada uma caixa e as regras utilizadas pela heurística favorecem a criação de colunas de caixas do mesmo tipo.

O algoritmo apresenta empacotamentos com alta eficiência, mesmo nos casos em que a carga a empacotar é fortemente heterogénea. [Gehring e Bortfeldt, 1997], descrevem uma metodologia onde são geradas combinações estáveis com as caixas, i.e., cada caixa é colocada na base do contentor ou em cima de outra caixa e o seu centro de gravidade deve ser sustentado pela caixa de baixo. É usado um algoritmo guloso que cria várias colunas verticais (colunas de caixas). Na construção das colunas, quando existem espaços livres, estes são preenchidos da seguinte forma: para um dado espaço e para um dado tipo de caixa, testam-se todos os posicionamentos da caixa no espaço através da rotação da caixa. Com base no volume da caixa e na melhor posição de encaixe desta no espaço livre é determinada a melhor rotação da caixa. A coluna que tenha maior volume total é incluída no conjunto de colunas. Em seguida as colunas são ordenadas decrescentemente segundo o valor da área da sua base, para ser efetuada a sua colocação dentro do contentor.

2.6.4. Empacotamento em Blocos

Existe uma outra forma de empacotamento para além das atrás descritas, na literatura verifica-se abordagens cujo empacotamento é feito em blocos de caixas. Os blocos podem ser constituídos por caixas do mesmo tipo, formando assim blocos homogéneos, tal como

ostentam [Bortfeldt e Gehring, 1998] e [Eley, 2002]. Caso contrário, são compostos por caixas de tipos diferentes, como é apresentado em [Davies e Bischoff, 1999], formando assim blocos heterogéneos.

[Bortfeldt e Gehring, 1998] assim como [Bortfeldt et al., 2003], expõem uma heurística gulosa para a construção de blocos homogéneos. Esta heurística garante também bons resultados para problemas fracamente heterogéneos. O algoritmo base aloca caixas num determinado contentor de uma forma cíclica, onde em cada iteração do ciclo é preenchido um único espaço no contentor. Para isto é selecionada a melhor combinação de um bloco ou par de blocos e respetivo espaço escolhido. Estas combinações são ordenadas de acordo com um ou dois esquemas de ordenação. A combinação bloco(s)/espaço é guardada numa lista de empacotamentos. Em seguida, é recomeçado um novo ciclo para preenchimento dos restantes espaços vazios.

2.6.5. Outras formas de empacotamento

Existem outras heurísticas que tem padrões de empacotamento que não estão limitadas às configurações de paredes, camadas, colunas ou blocos. Que podem ser encontradas em [Ngoi et al., 1994], [Faina, 2000] e [Bischoff, 2003].

Os primeiros, [Ngoi et al., 1994], propõem um procedimento baseado no sistema de representação espacial em matriz de [Ngoi e Whybrew, 1993]. Este sistema permite delinear um contentor com qualquer caixa e espaços vazios contidos por este. Em cada iteração do empacotamento é colocada uma única caixa e em seguida são identificadas todas as potenciais superfícies de empacotamento. Todas as caixas e todas as potenciais superfícies de empacotamento são comparadas e ordenadas. A combinação que tiver a maior ordem passa a ser a próxima caixa a empacotar. O esquema de ordenação é elaborado por forma a escolher caixas com dimensões tais que preencham o melhor possível o espaço livre.

[Faina, 2000], desenvolve um algoritmo onde a forma de empacotamento não está limitada ao empacotamento de paredes, camadas ou colunas, nem requer nenhuma regra de ordenação nem de prioridades. Este algoritmo é uma estratégia completamente tridimensional. O método geométrico é uma extensão para o caso tridimensional do método das zonas proposto em [Lipovetskji, 1988].

As caixas são alocadas no contentor pela ordem pela qual estas aparecem na lista de caixas para empacotamento, isto é, uma lista sem qualquer ordenação. A primeira caixa é colocada no contentor no seu canto inferior traseiro esquerdo. As outras vão sendo colocadas de acordo com a localização dada pelo método das zonas.

[Bischoff, 2003], desenvolveu uma heurística usando uma adaptação do sistema de representação matricial sugerido por [Ngoi et al., 1994]. Uma das diferenças é que não envolve a criação de uma matriz para a secção em cada um dos níveis de altura.

Neste caso, em vez de um grande número de matrizes só são necessárias duas: uma que representa a altura da superfície e outra que guarda a informação da capacidade de suporte da carga. Nesta abordagem só é colocada uma caixa de cada vez em cada iteração. O tipo de caixa, onde e como é que esta é posicionada no contentor, é definido através da análise de todas as possíveis combinações entre espaço livre e as caixas não empacotadas, com as respetivas orientações. As combinações admissíveis são avaliadas com base num esquema de pontuação, onde as opções são consideradas segundo cinco perspetivas diferentes. É selecionada a combinação que tiver melhor pontuação, sendo esta pontuação a soma de cada um dos valores dos cinco critérios

2.7. As primeiras Soluções

Na década de 60 muitos trabalhos importantes foram publicados sobre este tema, os trabalhos publicados por Gilmore e Gomory, [Gilmore e Gomory, 1961, 1963, 1965] tiveram a maior repercussão na comunidade científica e foram um dos que mais contribuíram para solucionar o problema. Em 1961, Gilmore e Gomory, [Gilmore e Gomory, 1961], formularam o Problema de Corte e Empacotamento Unidimensional e sugeriram a relaxação linear do problema, que pela primeira vez, resolveu problemas de grande porte para o caso unidimensional ultrapassando assim o obstáculo computacional dos PCE com inúmeros padrões de corte ou empacotamento.

Atualmente, os melhores métodos para aproximação de soluções são baseados nessa relaxação e muitas heurísticas são desenvolvidas para se obter uma solução inteira a partir da solução aproximada.

Dois anos mais tarde, os métodos expostos por Gilmore e Gomory em 1961, [Gilmore e Gomory, 1961], são estendidos e ajustados para um problema específico do PCE Unidimensional num estudo de um caso de corte do papel, [Gilmore e Gomory, 1963], e verificou-se que a abordagem funcionava bem na prática. Em 1965 Gilmore e Gomory, [Gilmore e Gomory, 1965], proporcionaram um grande avanço na forma como este problema era interpretado e observaram que empacotar caixas em contentores poderia ser visto como cortar pequenos paralelepípedos a partir de grandes paralelepípedos e apresentam assim um método para resolução de PCE para duas ou mais dimensões conhecido como método em dois Estágios.

Posteriormente na década de 70 os trabalhos de Haessler, [Haessler, 1971, 1975], apresentam um método de redução exaustiva repetida, que consiste em elaborar uma solução construindo e fixando os padrões um a um. Este método heurístico tem como objetivo adicional controlar o número de padrões na solução. Consideram-se, implicitamente, critérios adicionais relacionados com a qualidade da solução, tais como o número de movimentos ou reprocessamento de alguns itens, que o autor refere como não linearidades. Este método é computacionalmente eficiente para resolver o PCE unidimensional.

Até à década de 90, os desenvolvimentos sobre este problema concentraram-se essencialmente em processos heurísticos, sendo estes aplicados posteriormente em indústrias, que permitiram concluir que as soluções heurísticas eram bastante satisfatórias.

Em 1990 Goulimis, [Goulimis, 1990], apresenta uma abordagem de resolução exata no âmbito da indústria do papel. Goulimis utiliza uma formulação de colunas em programação linear inteira, considerando explicitamente os padrões que podem ser utilizados, onde o objetivo é a minimização do desperdício global e as restrições de satisfação da procura envolvem tolerâncias. Considera uma variável adicional que conta o total de rolos a utilizar e gera previamente um conjunto de padrões admissíveis. O problema é resolvido através de um algoritmo que combina uma fase de introdução de planos de corte de Gilmore e Gomory, [Gilmore e Gomory, 1963], seguida da aplicação de um algoritmo de pesquisa em árvore com ramificação e limitação com restrições sobre o valor das variáveis.

3. ALGORITMOS GENÉTICOS

Neste capítulo é feita uma exposição sobre algoritmos evolutivos, mais precisamente como surgiram, como funcionam e quais as características que distinguem cada um deles. Entre estes algoritmos, é dado maior ênfase aos algoritmos genéticos dado que fazem parte do objetivo principal deste trabalho.

3.1. Introdução

A atual tendência da sociedade de aproximar características humanas ou naturais a processos computacionais é cada vez mais uma prática observada. Existem vários fatores que determinam o interesse despertado por está técnica, mas a principal é a capacidade de obter soluções de problemas complexos.

Um dos ramos da ciência onde se destaca a aproximação dos processos naturais, aos processos computacionais é a computação evolutiva, uma vez que, tem por base os mecanismos evolutivos encontrados na natureza.

Estes mecanismos estão diretamente relacionados com a teoria da evolução de Darwin, onde este afirma que a vida na Terra é o resultado de um processo de seleção, feito pelo

meio ambiente, em que somente os mais aptos e adaptados possuirão mais pressupostos para sobreviver e consequentemente reproduzirem-se.

3.2. Teoria da seleção natural ou da evolução

Esta teoria foi publicada em 1859, na célebre obra "A origem das espécies" de Charles Darwin que marcou o início de uma revolução nas ciências e de uma profunda revisão nas conceções filosóficas e religiosas sobre a evolução dos seres. A validade desta teoria ainda é um assunto que incita grandes controvérsias, que vão do total apoio até a completa rejeição.

Para entender porque o darwinismo é mais do que apenas uma referência em biologia, é necessário abordar o pensamento pré-darwiniano, bem como alguns fundamentos da sua teoria. Por não ser este o foco desta dissertação, é feita apenas uma apresentação bastante sintética da teoria da evolução por seleção natural.

No século XIX, já existiam várias correntes de pensamento sobre as características dos seres vivos, em especial sobre a adaptação dos organismos e sobre a diversidade que existia na natureza. Os taxonomistas estruturavam essa diversidade de tal modo que, cada tipo de organismo combinava uma essência que seria definitiva e portanto eterna e imutável. Segundo Ernst Mayr, [Mayr, 1998], "O essencialismo sustenta que a diversidade da natureza, tanto inanimada como orgânica, é o reflexo de um número de universos imutáveis". Esta postura essencialista é uma das características do pensamento pré-darwiniano, e defende que em biologia, não existe uma proposta capaz de justificar toda a heterogeneidade da natureza.

Quando surgiu a teoria de Darwin esta representou o início de um contraponto ao essencialismo, que era o pensamento dominante da época.

No essencialismo a variação é entendida como a manifestação imperfeita das essências (ou tipos), no darwinismo demonstra-se a unicidade de cada coisa no mundo orgânico, fator que é fundamental para o entendimento dos fenômenos da seleção natural.

Darwin teve a oportunidade de conhecer e estudar uma enorme diversidade de fauna e flora durante a viagem a bordo do Beagle. Durante cinco anos, entre 1831 e 1836, percorreu

toda a costa sul-americana, parou nas ilhas Galápagos, passou pela Austrália e depois pelo Sul da África. Quando regressou escreveu alguns livros sobre a viagem e começou a desenvolver uma teoria que explica as suas observações, expressas na sua obra "A origem das espécies".

A noção de evolução tinha vindo a ser discutida à vários anos, mas é a compreensão do mecanismo de "descendência com modificação e da seleção natural" que possibilita a Darwin aprofundar a sua pesquisa e responder aos indícios que já apontavam para a evolução das espécies.

A ideia fundamental de Darwin é mostrar que o processo resultante das condições de sobrevivência em tempos de escassez de recursos leva necessariamente a que os indivíduos melhorem as suas características para enfrentar os mesmos problemas. Pelos seus termos:

"Considerando-se que, durante o longo percurso dos tempos e sob variáveis condições de vida, os seres vivos modificaram tanto diversas partes de seu organismo - e acho que isso é incontestável; considerando-se que, devido à alta tendência de crescimento geométrico do número das espécies, ocorre uma renhida luta pela sobrevivência, especialmente em determinada idade, ou determinada estação, ou determinados anos - e isso também certamente não tem contestação; consequentemente, dada a infinita complexidade das inter-relações dos seres vivos entre si e de cada um deles com suas as condições de existência, acarretando uma diversidade infinita quanto aos seus hábitos, estruturas e constituições internas, diversidade essa que lhes é proveitosa, penso que seria mesmo extraordinário se jamais houvesse ocorrido alguma variação útil exclusivamente para o bem-estar do ser, da mesma forma que ocorreram tantas variações úteis aos propósitos do homem. Mas se de fato ocorreram variações úteis a qualquer ser vivo, seguramente os indivíduos dotados delas terão maior probabilidade de ser preservados na luta pela existência; e em virtude do forte princípio de hereditariedade, eles tenderão a produzir descendentes dotados das mesmas características. Foi esse este princípio de preservação que, para ser conciso, dei o nome de Seleção Natural" [DARWIN, 2002].

É interessante notar como Mayr, [Mayr, 1998], apresenta a teoria de Darwin a partir de apenas três ilações baseadas em cinco factos:

1º. Todas as espécies possuem grande potencial de fertilidade, sendo que, se todos os indivíduos nascidos se reproduzissem, o tamanho das populações cresceria exponencialmente;

2º. O tamanho das populações é normalmente estável;

3º. Os recursos à disposição dessas populações são limitados e relativamente constantes;

1ª Ilação - Visto que é produzido um maior número de indivíduos do que os recursos disponíveis podem suportar e que o tamanho das populações permanecem estáveis, deve haver uma luta pela sobrevivência, com apenas uma pequena parte da descendência a sobreviver.

4º. Existe uma grande variabilidade nas populações, não existindo indivíduos que sejam idênticos;

5º. Grande parte da variação entre os indivíduos é herdável;

2ª Ilação – a sobrevivência não ocorre apenas ao acaso, mas depende em parte da constituição hereditária dos indivíduos que sobrevivem.

3ª Ilação – No fluxo das gerações, esse processo de seleção natural conduzirá a uma mudança gradual e continua das populações com a evolução e produção de novas espécies.

Darwin não pretende apresentar uma explicação para a origem da vida ou mesmo da primeira espécie, pois este inicia o seu trabalho considerando a presença de muitas espécies como um dado adquirido. Entende que o princípio de seleção natural é a visão unificadora para a adaptação e diversidade, ambos sendo aspetos diferentes de um único fenômeno complexo.

O que Darwin propõe é uma rutura com o pensamento essencialista. Ele sugere que um processo materialista e temporal num conjunto de passos sucessivos e sem propósito, pode produzir organismos que exibem algo que parece ser mais que apenas regularidade.

Lewontin,[Lewontin, 1984], afirma que esta é a principal mudança que o darwinismo introduz na biologia. Pelas suas palavras:

" [...] a natureza essencial da revolução darwiniana não é nem a introdução do evolucionismo como uma visão do mundo (já que historicamente este não é o caso), nem a ênfase na seleção natural como a principal força na evolução (visto que empiricamente este pode não ser o caso), mas sim a substituição de uma visão metafísica da variação entre os organismos por uma visão materialista"

Após a morte de Darwin, em 1882, surgiram vários pensamentos sobre biologia que se apoiavam na teoria de Darwin.

Mais tarde com a redescoberta das leis de Mendel, em 1900, levou a polarização da discussão entre os apoiantes de Mendel e os naturalistas. Os últimos interessavam-se pela diversidade enquanto os geneticistas que apoiavam Mendel procuravam explicar a transformação a partir do foco nos genes e nas suas características associadas. Segundo Ernst Mayr, [Mayr, 1998], apenas a partir de uma nova geração de geneticistas, entre 1936 e 1947, foi possível uma síntese entre estas duas tradições de pesquisa, possibilitando o estabelecimento de uma base comum à biologia, afirmando o gradualismo da evolução e robustecendo o pensamento populacional.

Em resumo, a proposta de evolução por seleção natural da síntese neodarwiniana pode ser entendida a partir dos seguintes atributos apresentados pelos organismos: herança das características dentro de uma história populacional, variação individual nas características, tendência dos organismos em aumentar seu número exponencialmente com a consequente competição por recursos limitados num ambiente em constante mudança. Dadas estas condições, segue-se uma alta probabilidade de que alguma variação numa característica torne o seu possuidor mais apto e que este tenha uma maior probabilidade de sobrevivência e assim tenda a deixar um número proporcionalmente maior de descendentes com a mesma variação nas gerações seguintes. Em resumo, qualquer ser que apresente variações, que se reproduza e herde características pode evoluir. Este é o modelo causal que explicaria a fixação de certas características vantajosas nas populações.

Este modelo resolveu uma série de problemas em biologia, possibilitando um novo significado para conceitos como evolução e espécie, criando novas possibilidades para entendermos como os fenômenos evolutivos efetivamente operam nos casos individuais.

3.3. Algoritmos Evolutivos

Uma área da computação evolutiva são os algoritmos evolutivos que são processos metaheurísticos, que tentam imitar comportamentos da evolução natural, de forma a solucionarem problemas que necessitem de adaptação, pesquisa e otimização. São métodos estocásticos e iterativos, não asseguram a convergência para a melhor solução uma vez que se baseiam em métodos de aproximação. Estes algoritmos trabalham com um conjunto de indivíduos ou população, onde cada indivíduo representa uma potencial solução do problema de otimização.

O processo evolutivo é iniciado por gerar, aleatoriamente ou usando métodos específicos para esse efeito, a primeira população denominada por população inicial, depois segue-se a avaliação de cada solução da população através de uma função de aptidão. A função de aptidão classifica cada uma das soluções da população denominada por valor de aptidão ou *fitness*, esta classificação mede a qualidade da solução do problema em estudo e é utilizada para orientar a pesquisa de novas soluções.

O processo evolutivo segue, aplicando às soluções da população que podem pertencer ou não à população inicial, um procedimento para gerar uma nova população constituído pelas quatro etapas principais: seleção, cruzamento, mutação e substituição. A seleção consiste no processo de criar uma população provisória com algumas das soluções da população principal. As soluções com maior valor de aptidão têm maior possibilidade de estarem presentes na população provisória do que as soluções com menor valor de aptidão. A estas soluções é então aplicado o operador genético cruzamento seguido de mutação, de forma a gerar uma nova população. A seguir, as soluções da população principal são permutadas pelas novas soluções, fazendo com que, normalmente, as soluções com melhores valores de aptidão se conservem e as com os piores valores de aptidão sejam excluídas. Este processo só termina, geralmente, quando for atingido um número predefinido de gerações, ou quando não for detetada uma melhoria significativa nas últimas gerações.

Existem vários tipos de algoritmos evolutivos, devendo-se a classificação destas técnicas a fatores históricos relacionados com o aparecimento de diferentes linhas de investigação. No entanto, as técnicas mais conhecidas são as Estratégias Evolucionárias, a Programação Evolucionária e os Algoritmos Genéticos.

3.3.1. Origem dos Algoritmos evolutivos

As primeiras experiências de associar a evolução natural, a problemas de otimização surgiram, na década de 50, dando origem aos primeiros algoritmos evolutivos. Uma destas experiências foi a criação de um modelo discreto "autómatos celulares" que, servia para estudar processos de crescimento e auto-reprodução, este modelo foi criado por John von Neumann's em Princeton [John, 1952]. Em 1953, Nils Barricelli, [Barricelli, 1954], desenvolveu um programa que, simulava um ambiente separado por células numa grelha. O objetivo era preencher algumas células com números e fazer estes migrarem, para as células vizinhas, com base num conjunto de regras. Ao definir um padrão de propagação, dois números quando colidiam na mesma célula, competiam pela sobrevivência.

Nesta experiencia Barricelli descobriu que, mesmo com regras muito simples para os números se propagarem por todo o ambiente, determinados padrões de números iria evoluir e só poderia persistir quando os padrões de outros também estavam presentes algo semelhante a simbiose.

Em 1957 Alex Fraser, um geneticista britânico, publicou uma série de artigos sobre a experiencia em diferentes meios [Alex, 1957], da simulação de organismos com recombinação sexual entre múltiplas características, mensuráveis e em meios diferentes. A forma comum de recombinação era um operador de crossover, que tirava partes de uma solução e juntava-as a partes de uma outra solução.

Esta forma de recombinação foi mais tarde questionada no trabalho de Ingo Bremermann, [Ingo, 1962], onde sugere que através de uma recombinação ou um optimizador evolutivo contendo elementos de mutação e cruzamento, é possível gerar uma pesquisa mais robusta e mais eficiente mesmo em sistemas complexos.

Ainda na mesma década em 1964, os investigadores Alemães: Ingo Rechenberg, Hans Paul Schwefel e Peter Bienert conceberam um sistema evolutivo para otimização de formas

aerodinâmicas, que pode ser considerada a primeira aplicação de princípios biológicos à solução de problemas de projeto em engenharia, [Ingo e Hans Peter, 1964].

Mais tarde em 1966 as pesquisas desenvolvidas por Lawrence J. Fogel nos Estados Unidos [Fogel et al, 1966], levaram ao aparecimento de técnicas para a geração de inteligência artificial a partir, da evolução de máquinas de estados finitos surgindo então a Programação Evolucionária.

3.3.2. Programação Evolucionária

Esta técnica surge em [Fogel et al, 1966], com a finalidade de gerar máquinas de estados finitos, para tarefas de pré-edição de sequências de símbolos. Fogel propôs a utilização dos próprios mecanismos evolutivos para se alcançar máquinas capazes de ter comportamento inteligente. Atualmente, estes algoritmos são utilizados sobretudo em problemas de otimização numérica.

A característica principal da Programação Evolucionária refere-se à forma como os descendentes são gerados sendo o operador genético mutação o único responsável pela geração de descendentes. Este operador é definido conforme o problema e está sujeito a adaptações durante o processo evolutivo.

Dado que não existe o operador genético de cruzamento, é possível usar qualquer representação para as soluções (está dependente do tipo de problema que se queira resolver), desde que se defina um operador mutação adequado ao problema. Na versão original, por exemplo, cada solução era uma máquina de estados finitos representada por um grafo. Em problemas de otimização numérica, as soluções são representadas por vetores de valores reais, como por exemplo o problema do caixeiro-viajante, onde são utilizadas listas ordenadas.

Na Programação Evolucionária é simulada a evolução natural através de um processo fundamentado nas soluções, enfatizando o comportamento na ligação entre os progenitores e os descendentes, em vez da ligação genética, como acontece com os restantes Algoritmos Evolutivos.

3.3.3. Estratégias Evolucionárias

Ainda na mesma década [Rechenberg, 1964] iniciou o estudo das Estratégias Evolucionárias.

Esta técnica foi desenvolvida por Rechenberg, [Rechenberg, 1973], com base no seu trabalho em 1964, [Rechenberg, 1964], com a finalidade de resolver problemas de otimização. Posteriormente foram desenvolvidos novos esquemas evolutivos, mas seguindo os mesmos princípios, por Schwefel [Rechenberg, 1981]. Estas técnicas foram desenvolvidas para serem aplicadas a problemas de engenharia.

A primeira versão que surgiu desta técnica utiliza apenas duas soluções, uma progenitora e uma descendente, as quais são codificadas usando uma representação decimal. Esta versão considera apenas o operador genético mutação e um mecanismo de seleção e não considera o operador cruzamento. Este tipo de Estratégias Evolucionárias designa-se por EE- (1+1), na atual notação de Estratégias Evolucionárias [Bäck et al., 1991].

Entretanto surgiram, algumas generalizações desta técnica, nomeadamente a sua aplicação quando existem mais do que uma solução, denominadas por EE- (μ+1). Nesta variante consideram-se μ soluções progenitoras, das quais é selecionada uma delas para a geração de uma solução descendente.

Mais tarde, desenvolveram-se outras variantes, designadas por EE- (μ+λ) e EE- (μ,λ), onde é gerado por mutação λ descendentes a partir de μ progenitoras e aplicados esquemas de seleção diferentes para escolher as μ melhores para a geração seguinte.

No método EE- (μ+λ) as soluções da população para a geração seguinte são escolhidas entre as μ progenitoras e as λ descendentes. No método EE-(μ,λ) as soluções são escolhidas unicamente das λ descendentes (assumindo que $\lambda > \mu$).

Apesar das Estratégias Evolucionárias originais usarem apenas o operador genético mutação para gerar novos indivíduos, foi introduzido, posteriormente, o operador genético cruzamento, ao qual foi aplicado simultaneamente com a mutação [Schwefel, 1995].

As Estratégias Evolucionárias apresentaram-se algoritmos de otimização consistentes e eficientes, não exigindo nenhuma condição relativa à continuidade e convexidade do espaço de pesquisa, ao contrário de outros algoritmos de otimização [Schwefel, 1995].

3.4. Algoritmos Genéticos

Na década de 70 surgiram os Algoritmos Genéticos (AGs) e tornam-se o grupo dos Algoritmos Evolutivos mais utilizados na resolução de problemas de pesquisa e de otimização. Estes algoritmos são métodos adaptativos inspirados nos mecanismos de evolução de populações de seres vivos, aos quais evoluem de acordo com os princípios da seleção natural, descritos pela primeira vez por Charles Darwin.

Os princípios básicos dos AGs foram apresentados por Holland, [Holland, 1975] e desenvolvidos por ele e pelos seus colaboradores (alunos e colegas) na Universidade de Michigan, nas décadas de 1960 e 1970. Os objetivos da pesquisa de Holland foram os seguintes [Goldberg, 1989]:

a) abstrair e apresentar rigorosamente os processos de adaptação dos sistemas naturais,
b) criar software para sistemas artificiais que detenham os mecanismos importantes dos sistemas naturais.

Contudo, foram os trabalhos desenvolvidos por um dos alunos de Holland, David Goldberg, que tornaram estes algoritmos populares, quando este os aplicou para resolver um problema na área de controlo de transporte de gás natural em gasodutos (no âmbito da sua tese de doutoramento). O algoritmo genético original, proposto por Holland, designa-se por AG canónico ou simples [Goldberg,1989].

Os AGs diferem dos mecanismos clássicos de otimização e de pesquisa, nos seguintes pontos segundo [Goldberg, 1989]:

1. utilizam uma codificação do conjunto de parâmetros e não os próprios parâmetros;
2. procuram a partir de uma população de vários pontos e não de um único ponto;
3. usam a informação relativa retornada pela função objetivo, e não de derivações ou de outro tipo de conhecimento auxiliar;
4. usam regras de transição probabilísticas, e não determinísticas.

É comum, na terminologia dos AGs, utilizar-se diversos sinónimos para os mesmos elementos pode-se usar-se qualquer um dos termos para solução, fenótipo ou indivíduo para se referir os pontos do espaço das soluções (também referido como espaço do fenótipo). No contexto dos AGs, pode-se utilizar-se qualquer um dos termos genótipo, cromossoma e também indivíduo para referir os pontos do espaço onde se concretiza a pesquisa evolucionária (também referida por espaço do genótipo). Os elementos dos indivíduos (lugar e objeto) também podem ter vários sinónimos: um lugar designa-se geralmente por variável, locus (plural: loci), posição, ou gene; um objeto num lugar pode-se chamar de valor ou alelo.

É comum utilizar-se os termos solução e indivíduo (no espaço dos genótipos) como sinónimos, dado que um indivíduo é uma representação da solução.

Neste trabalho os termos solução e indivíduo são utilizados como sinónimos, onde cada solução/indivíduo é composta por um conjunto de variáveis ou genes, cada uma com um certo valor.

Os AGs utilizam uma população de indivíduos (que representam as soluções do problema), à qual são aplicados mecanismos de seleção e de substituição, e operadores genéticos cruzamento e mutação, originando gerações sucessivas de indivíduos cada vez mais aptos. A pesquisa de uma nova solução é direcionada apenas pelo valor de aptidão de cada indivíduo da população: os indivíduos com maior valor de aptidão terão maiores possibilidades de serem selecionados para se reproduzirem (passando a informação genética para os descendentes), do que aqueles que apresentam menor valor de aptidão (cuja informação genética tende a desaparecer). As características essenciais dos AGs são as seguintes:

- utilizam uma população de soluções do problema;
- a cada solução é associado um valor de aptidão que refere a sua capacidade de sobrevivência e de reprodução;
- as soluções que apresentam os melhores valores de aptidão têm mais probabilidades de se cruzarem com outras soluções da população;
- durante a reprodução são geradas novas soluções que herdam determinadas características dos seus progenitores;

- através da combinação de algumas boas características dos progenitores é possível obter uma geração com melhores soluções.

A estrutura de um AG pode ser descrita como mostra a Figura 9:

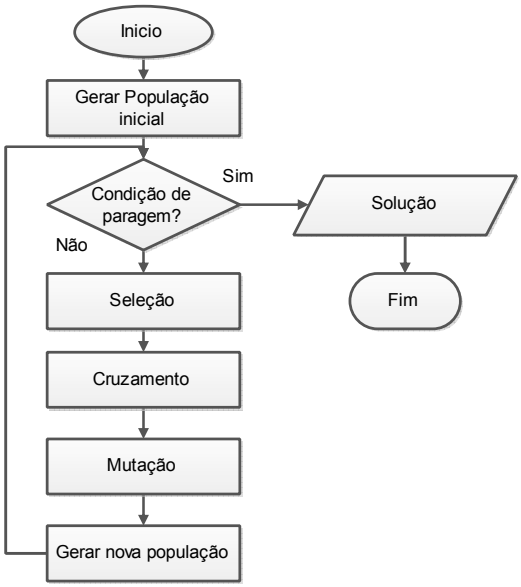


Figura 9 Funcionamento do Algoritmo Genético

Durante o processo de desenvolvimento de um AG para solucionar um certo problema geralmente é necessário considerar a representação (codificação) das soluções e definir a função aptidão, dado que são as únicas entidades que dependem do problema. Assim sendo, é essencial considerar o processo de um AG com os seguintes aspetos: representação (codificação) das soluções, valor de aptidão, início do processo, população, mecanismos de seleção, operadores genéticos (cruzamento e mutação), mecanismos de substituição e critérios de paragem.

3.4.1. Codificação e representação dos indivíduos

Para se definir um AG é preciso inicialmente estabelecer uma relação entre o contexto do problema original e o espaço de resolução do problema onde surge o processo de evolução. Este primeiro passo é normalmente designada por representação, que significa especificar uma conexão entre um conjunto de soluções do problema original (espaço dos fenótipos) e um conjunto de indivíduos (no espaço dos genótipos) [Eiben e Smith, 2003]. Ou seja, a representação consiste em determinar uma codificação apropriada para as possíveis

soluções do problema (fenótipos) ao nível das características genéticas de cada indivíduo (genótipos). A escolha de um tipo de representação que simplifique o tratamento das soluções do problema por parte do AG necessita, habitualmente, de um enorme conhecimento das particularidades do problema.

Contudo, os termos de representação podem ser utilizados nos dois sentidos [Eiben e Smith, 2003] como codificação e como descodificação. A codificação refere-se à representação de um fenótipo no espaço dos genótipos (por exemplo, representação ou codificação binária das soluções). A descodificação está relacionada com a ligação inversa, isto é, do genótipo para o fenótipo, a qual deve precisar que para cada indivíduo exista no máximo uma solução correspondente.

A representação mais comum, e que coincide com a ideia original de Holland, é aquela que utiliza o alfabeto binário para codificar os valores possíveis de cada variável do indivíduo. No entanto, outras representações podem ser usadas, as quais devem ser apropriadas às características do problema em estudo, tais como: números reais, números inteiros e até mesmo caracteres.

A Figura 10 mostra um exemplo da representação de um cromossoma

Figura 10 Exemplo de representação de um cromossoma

3.4.2. Início do processo

A primeira fase do processo do algoritmo genético tem como objetivo criar a primeira população (também denominada de população inicial), onde indivíduos podem ser ou não gerados aleatoriamente.

É importante que exista nesta população uma grande variedade de indivíduos, de forma a possibilitar uma exploração mais abrangente do espaço de pesquisa. Para tal, pode ser utilizado um gerador de valores aleatórios de distribuição uniforme, ou então, utilizar uma metodologia construída para o efeito.

Existem outras formas de criar uma população inicial como heurísticas específicas para o problema que podem criar uma população inicial de indivíduos com elevada aptidão [Eiben e Smith, 2003]. Contudo, esta estratégia pode implicar um grande esforço computacional, mas pode trazer vantagens, se acelerar a convergência do algoritmo ou evitar que esta seja atingida prematuramente.

Esta fase é um passo muito relevante na construção de um AG, uma vez que uma população inicial mal construída pode levar o AG a dirigir-se prematuramente, ficando retido em ótimos locais. Esta situação pode acontecer caso a informação genética guardada nos indivíduos de uma população não tiver a informação essencial para melhorar a qualidade da solução.

3.4.3. Função de avaliação (aptidão)

A função de avaliação é o processo do AG onde são classificadas as condições de adaptabilidade dos indivíduos da população, representa o suporte ao mecanismo de seleção e simplificando, assim, o melhoramento da população (ou seja, definindo o que significa melhoramento). Tecnicamente, a função de avaliação é um processo onde é atribuído uma medida de qualidade aos indivíduos (genótipos). Geralmente, esta função é formada por uma métrica de qualidade para o espaço das soluções possíveis (dos fenótipos) e pela representação inversa (descodificação): por exemplo, o valor de aptidão do genótipo 01101 é f(32), em que f é a função a otimizar [Eiben e Smith, 2003].

A função da aptidão atribui um valor de aptidão a cada indivíduo (genótipo), sendo este elemento essencial para o processo de seleção. Este cálculo do valor de aptidão dos indivíduos é feito utilizando, geralmente, uma função de avaliação específica do problema em questão. A função de avaliação atribui, a cada indivíduo, um valor numérico que classifica o potencial da solução. É de notar que um valor de aptidão pode estar relacionado a mais do que uma solução (fenótipo).

Ao definir-se uma função de aptidão, é extramente importante que esta seja apropriada ao problema para uma execução correta do AG, dado que a função de aptidão representa o ambiente do problema. Idealmente procura-se que esta função de aptidão tenha um comportamento suave e regular, procurando assim atribuir a indivíduos semelhantes e com

as mesmas características (que tenham algumas características em comum), valores de aptidão muito próximos.

3.4.4. População

Uma população representa um conjunto de indivíduos (genótipos), com a capacidade de permitir a existência de elementos iguais. A definição de uma população pode representar apenas a especificação do número máximo de indivíduos que esta suporta, ou seja, o seu tamanho.

O tamanho das populações é uma configuração relevante para a implementação de AG, dado que este valor vai interferir, quer a qualidade das soluções, quer o tempo de processamento do algoritmo. Populações pequenas têm como resultado uma fraca variedade genética nos seus elementos, o que provoca uma menor cobertura do espaço das soluções e, consequentemente, à convergência prematura (ótimos locais). Populações grandes oferecem uma maior variedade de soluções na população, devido a uma maior cobertura do espaço das soluções, permitindo assim prevenir as convergências prematuras, mas esta vantagem é conseguida através de um maior esforço computacional. Os AGs podem utilizar populações de todos os tamanhos, sendo normal este tamanho ser constante (não se alterando durante o processo de evolução).

Uma característica indispensável para o bom funcionamento de um AG é a existência de uma grande variedade de indivíduos na população. Esta variedade é uma medida associada à diferenciação das soluções no espaço das soluções. Se o nível da variedade da população for baixo, oferece mais garantias que os indivíduos são muito semelhantes, levando o operador genético cruzamento a perder alguma da sua capacidade de troca de informações úteis entre os indivíduos da população, podendo provocar uma lenta progressão da pesquisa ou estacionar. Se o nível da variedade da população for alto, isto possibilitará explorar melhor o espaço de pesquisa, o que é importante para evitar locais ótimos.

Esta necessidade de controlar a variedade populacional implica que as populações tenham tamanhos finitos e não demasiado grandes. Caso se verifique que, o AG tem uma população absurdamente grande pode provocar um esforço computacional muito elevado levando à inaplicabilidade prática do AG.

Contudo não existe um valor único para a variedade populacional pois, geralmente, pode referir-se à gama dos valores de aptidão, ao número de soluções (fenótipos) diferentes ou ao número de indivíduos (genótipos) diferentes [Eiben e Smith, 2003].

3.4.5. Mecanismos de seleção

A seleção é o mecanismo que se baseia no princípio de Darwin que defende a "sobrevivência dos mais aptos". Segundo este princípio os indivíduos que apresentam melhores valores de aptidão têm uma probabilidade maior de serem selecionados para reprodução. Este mecanismo permite determinar quais os indivíduos (progenitores) que devem ser escolhidos para o cruzamento e os indivíduos (descendentes) que devem sobreviver para a geração seguinte.

Existem dois tipos de mecanismos de seleção: baseados na proporção dos valores de aptidão e baseados na ordenação dos valores de aptidão. No mecanismo da seleção baseada na proporção, os indivíduos são selecionados através dos seus valores de aptidão e da ligação com os valores de aptidão dos outros indivíduos da população. No mecanismo da seleção baseada na ordenação, os indivíduos são selecionados através das respetivas posições na população ordenada e não através dos seus valores de aptidão.

Na seleção baseada na proporção existem dois métodos que são os mais utilizados em problemas de otimização, são o método da roleta e a amostra universal estocástica. O método da roleta [Goldberg, 1989] é um método estocástico que consiste em associar os indivíduos a frações contíguas de uma roleta, onde cada fração é proporcional à aptidão do indivíduo que lhe está associado (os indivíduos com maior valor de aptidão têm maiores probabilidades de serem escolhidos). A Figura 11 ilustra um exemplo da associação através do método da roleta.

Cromossoma	Aptidão	%
1	10	50
2	5	25
3	2,5	12,5
4	2,5	12,5

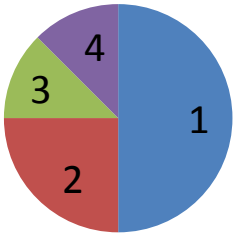


Figura 11 Exemplo do método da roleta

Durante o período de execução deste método existem vários lançamentos da roleta, e em cada lançamento são selecionados os indivíduos que apresentam maiores frações atribuídas pelo lançamento. Contudo, este mecanismo apresenta duas desvantagens: redução da variedade e convergência prematura (os mais aptos podem ser selecionados muitas vezes), e estagnação da população (ao fim de algumas gerações, o valor médio da aptidão pode ser muito próximo dos melhores valores de aptidão).

No método denominado por amostra universal estocástica [Baker, 1987], os indivíduos são associados a porções proporcionais aos seus valores de aptidão e contíguas de uma roleta, tal como no método da roleta. Este método difere do anterior dado que a roleta tem N ponteiros equidistantes (N é o número de indivíduos a selecionar) e gira apenas uma vez, sendo escolhidos os indivíduos marcados pelos N ponteiros (assim, os indivíduos menos aptos têm mais possibilidades de serem escolhidos do que no método anterior).

Entre os mecanismos de seleção baseados na ordenação os mais utilizados, estão a ordenação linear, a seleção por truncatura e a seleção por torneio.

No mecanismo denominado por ordenação linear [Baker, 1985], é feita uma ordenação dos indivíduos da população conforme os seus valores de aptidão, sendo atribuído a cada indivíduo um valor mediante a sua posição na população ordenada. Ao pior indivíduo é dado o valor 1 e ao melhor o valor POP (em que POP é o tamanho da população).

Depois, a cada indivíduo é dada uma probabilidade de seleção calculada com base numa dada distribuição (as mais utilizadas são a linear e a exponencial). Esta técnica evita a convergência prematura do algoritmo (não existe favorecimento dos melhores indivíduos) e impede, a estagnação da população em gerações seguintes.

A seleção por truncatura é o mecanismo onde, é feita uma ordenação dos indivíduos de acordo com os valores de aptidão respetivos e, de seguida, são eleitos aqueles indivíduos cujos valores de aptidão sejam maiores do que um limiar predefinido (só são selecionados os melhores indivíduos).

O mecanismo de seleção por torneio [Goldberg, 1989], consiste em eleger aleatoriamente um respetivo número de indivíduos da população (denominado por dimensão do torneio) e fazer um torneio entre eles. Um torneio consiste em comparar os valores de aptidão dos indivíduos envolvidos, sendo o selecionado aquele que melhor valor de aptidão apresenta.

O número de torneios que são efetuados é igual ao número de indivíduos a serem selecionados, ou seja, igual ao tamanho da população. Esta técnica não conduz à convergência prematura (desde que a dimensão dos torneios seja pequena), evita a estagnação da população, é fácil de implementar e não necessita de grande esforço computacional. Este é talvez o mecanismo de seleção mais utilizado na resolução de problemas de otimização.

Uma técnica que está internamente relacionada ao mecanismo de seleção é o elitismo. O elitismo foi proposto por [De Jong, 1975] e consiste em conservar na população os seus melhores indivíduos, os quais são selecionados diretamente para a próxima geração.

Vários investigadores têm visto no elitismo vantagens significativas para o desempenho dos AG [Mitchell et al., 1997]. Com esta técnica, procura-se, por um lado, assegurar que os melhores indivíduos de cada geração não sejam eliminados pela ação dos operadores genéticos cruzamento e mutação (a definir mais à frente), e por outro, acelerar a convergência do algoritmo.

Uma desvantagem que o elitismo apresenta está no facto de os melhores indivíduos encontrados serem os mesmos durante varias gerações consecutivas, esta desvantagem pode implicar a presença de várias imitações dos mesmos indivíduos em cada geração. Desta forma, pode haver perda de variedade entre os indivíduos da população e direcionar a pesquisa para um ótimo local. Assim sendo, e para contornar este efeito negativo, o elitismo não deve ser usado constantemente, mas sim esporadicamente durante o processo evolutivo.

O número de indivíduos que originam a elite compõe um dos critérios mais importantes dos AGs, pois se este valor não for apropriado pode prejudicar a convergência do AG. Geralmente, este é um número baixo (no máximo 10% da população) e a sua amostragem pode ser direta (os melhores) ou por sorteio (os melhores entre os melhores da população).

3.4.6. Operador genético cruzamento

O operador genético cruzamento (ou recombinação) é o processo onde é efetuado trocas de genes entre dois indivíduos. Neste processo são criados dois novos indivíduos (descendentes), obtidos da combinação das características contidas num par de indivíduos (progenitores). O êxito do AG está apoiado na expectativa de que o resultado do cruzamento entre indivíduos (progenitores) com melhores valores de aptidão crie novos indivíduos (descendentes) supostamente com melhores características (relativamente aos progenitores).

É fundamental que o operador cruzamento proporcione a propagação ao longo das gerações, das características genéticas dos melhores indivíduos, para conseguir explorar bem o espaço das soluções, assim como agrupar as melhores características genéticas num indivíduo que apresentará a solução final do problema de otimização em questão.

Segundo [Baker, 1985] existem diversos tipos de cruzamentos, cada cruzamento depende do tipo de representação utilizada na codificação dos indivíduos, os tipos de cruzamento mais utilizados são os que a seguir se descrevem.

O cruzamento com um ponto de corte consiste em separar em duas partes as características de dois individuo através de um ponto de corte determinado aleatoriamente. Desta forma, são obtidas quatro partes de genes de dois indivíduos selecionados, sendo depois cruzadas nos descendentes, que recebem uma sequência de cada um dos progenitores. A Figura 12 ilustra um exemplo de um cruzamento de um ponto.

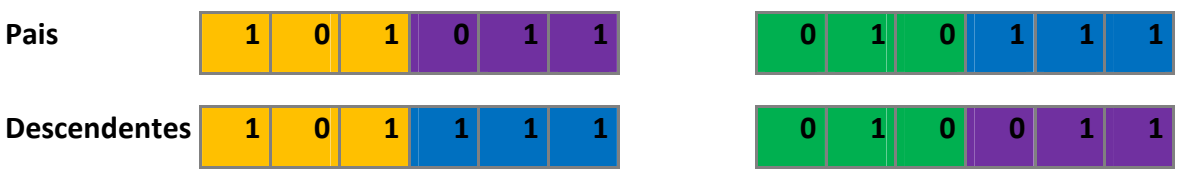


Figura 12 Cruzamento de um ponto

O cruzamento com dois pontos de corte é feito através de dois pontos de corte, estes cortes são determinados aleatoriamente em duas posições de um indivíduo, de seguida os dois individuo são separando em três, através dos mesmos cortes. Desta forma, são formadas seis sequências de genes dos indivíduos selecionados, em que as sequências do meio são trocadas nos descendentes como ilustra a Figura 13.

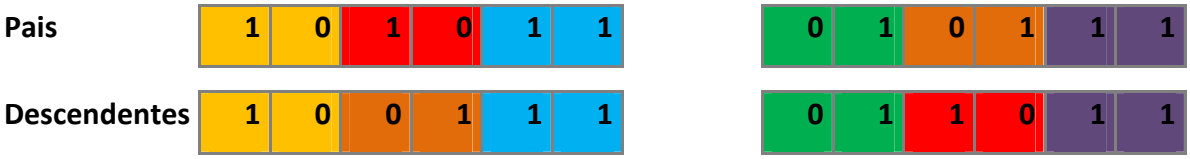


Figura 13 Cruzamento de dois pontos

O cruzamento com vários pontos de corte é feito de forma análoga ao cruzamento anterior, mas existe mais do que dois pontos de corte, onde cada um dos descendentes recebe de um dos progenitores as sequências de genes de índice ímpar e do outro as sequências de genes de índice par, e o outro descendente recebe as restantes. A Figura 14 ilustra um exemplo de um cruzamento com vários pontos.

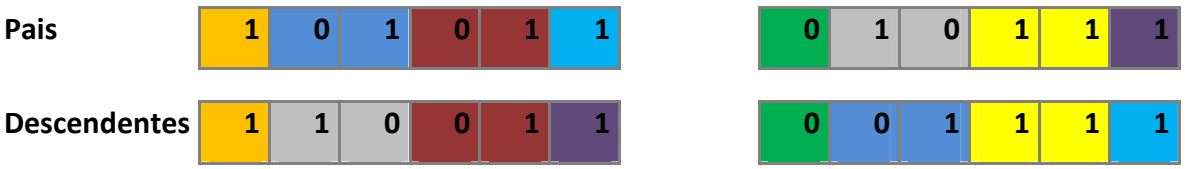


Figura 14 Cruzamento com vários pontos

O cruzamento uniforme é um mecanismo mais elaborado e consiste na aplicação de uma máscara binária, gerada aleatoriamente e de tamanho igual ao dos indivíduos. Esta máscara vai dizer qual ou quais os genes que o progenitores herda dos pais. Para as posições da máscara com valor 0 herda de um pai, e do outro pai para as posições da máscara com valor 1, o mesmo acontece com o outro descendente, mas agora para os valores da máscara trocados como mostra a Figura 15.

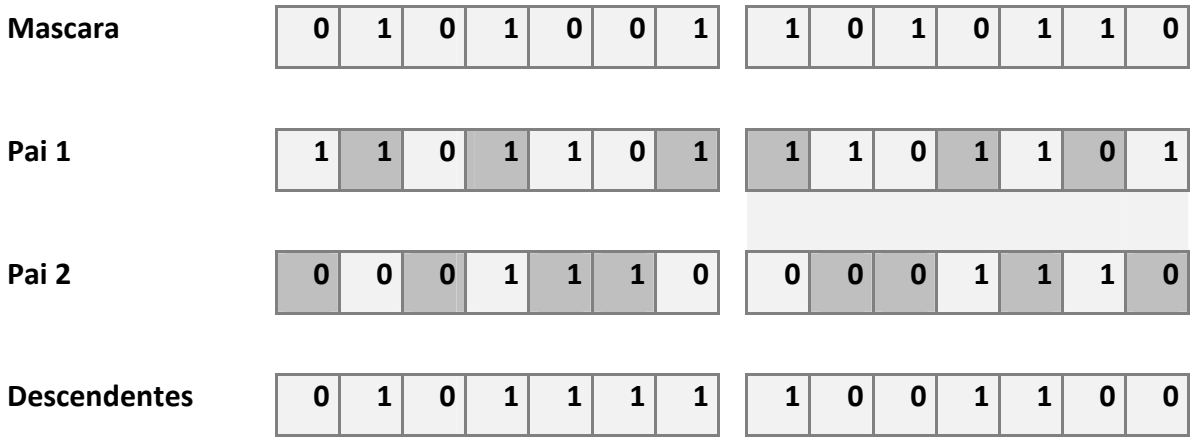


Figura 15 Cruzamento uniforme

Este tipo de cruzamento pode apresentar algumas restrições na sua aplicabilidade nalgumas representações, como por exemplo nas representações baseadas em permutações utilizadas normalmente em problemas como o caixeiro-viajante. Nestes casos, é indispensável utilizar outros cruzamentos.

O desempenho obtido por cada um dos tipos de cruzamento depende muito das características dos problemas que se pretende resolver.

Na utilização deste operador existe um aspeto importante a ter em conta que se deve à atribuição de uma probabilidade de cruzamento. Esta probabilidade representa a possibilidade dá aplicação do operador cruzamento a um dado par de indivíduos. Para valores, maiores existe a possibilidade de surgirem novos indivíduos na população e em maior número. Verifica-se que as probabilidades mais utilizadas variam entre 0.5 e 1.0.

3.4.7. Operador genético mutação

O operador genético mutação representa a alteração aleatória de um valor de um dos genes de um indivíduo. Procura-se com este operador genético nos AGs criar novos genes que não se encontravam nas populações anteriores.

Estes novos genes vão substituir genes selecionados aleatoriamente de um individuo, que na prática representa a alteração do valor que se encontra nas posições determinadas aleatoriamente no individuo por um outro valore do conjunto associado à representação usada, caso exista mais do que um valor possível para substituir, este é escolhido aleatoriamente.

Em representações baseadas em permutações, uma forma simples para este operador atuar consiste em comutar a posição dois elementos.

Existem outros tipos de mutação que têm sido propostas, como a mutação múltipla que propõem aplicar o operador genético a todos os genes do indivíduo. Neste tipo de mutação, podem ser provocadas diferenças significativas entre o indivíduo antes e depois da mutação, para o espaço de pesquisa, traz vantagens dado que evita o ótimo local e explora melhor as soluções possíveis.

Um outro tipo de mutação é a localizada, onde só são afetados pela mutação os genes menos significativos do indivíduo, levando a pequenas perturbações nos indivíduos.

Na aplicação deste operador existem um aspeto importante a considerar que é a taxa de atuação sobre o indivíduo, ou seja, a probabilidade de mutação. Segundo alguns estudos apontam que uma probabilidade elevada tenderá a tornar o AG num algoritmo principalmente aleatório. Para evitar este comportamento o valor da probabilidade de mutação deve ser baixo, o suficiente para variar os indivíduos da população e não afetar a convergência do algoritmo. Em alguns casos, pode verificar-se que a probabilidade de mutação pode depender de uma função matemática monotonamente decrescente, levando a mutação a ocorrer com maior frequência durante as primeiras gerações (quando é necessário uma pesquisa mais alargada) e mais excecionalmente nas últimas gerações (para possibilitar a convergência do algoritmo) [Reeves, 1995] e [Gomes et al. 2004].

3.4.8. Mecanismos de substituição

Quando terminado o processo de geração e avaliação através da função de aptidão, os indivíduos são inseridos na população. Para isso é necessário definir quais os indivíduos descendentes que serão introduzidos na população da nova geração, e quais os indivíduos da população atual que devem ser substituídos. Para tal, determina-se a taxa de substituição, que vai indicar qual a proporção de indivíduos da população que vão ser substituídos a cada geração. Para valores menores, menor será a diferença genética entre gerações, o que provocará um abrandamento da convergência do algoritmo.

Existem vários mecanismos de substituição, entre estes o mais utilizado é aquele que consiste em gerar um número de descendentes iguais ao número de indivíduos da

população, para a seguir, substituir todos estes indivíduos pelos seus descendentes que foram criados. No entanto, existem outras formas como, por exemplo, o elitismo onde, é feita a escolha dos melhores indivíduos da população os quais são selecionados imediatamente, e sem qualquer tipo de comparação, para a população da geração seguinte.

Outro mecanismo de mutação, é o AG de estado estacionário, que consiste em selecionar apenas dois progenitores da população atual para reprodução (cruzamento seguido de mutação), da qual resulta apenas um ou dois novos indivíduos [Whitley et al, 1989]). Estes novos indivíduos criados vão substituir outros indivíduos da população escolhidos de acordo com o seu valore de aptidão, podendo ser escolhidos os que têm piores valores de aptidão ou aleatoriamente. Contudo, por vezes, a substituição só é feita caso os novos indivíduos tenham valores de aptidão melhores do que os indivíduos selecionados para serem substituídos.

O último mecanismo consiste em escolher uma determina percentagem pré-definida da população corrente para reprodução [De Jong e Sarma, 1992].

Analisando estes mecanismos verifica-se que o primeiro difere dos outros três devido ao facto, de nestes três mecanismos haver a hipótese de competição entre indivíduos de gerações diferentes, dado que numa geração existem sempre indivíduos de gerações anteriores, o que não acontece no primeiro mecanismo, em que todos os indivíduos da população corrente são substituídos por novos indivíduos.

3.4.9. Condições de paragem

O processo de evolução pode ser terminado devido às características do problema em estudo ou ao tempo de execução do algoritmo. Desta forma, é possível definir a qualidade da solução a encontrar, de acordo com o tempo e os recursos disponíveis. A condição de paragem do processo de evolução deve ter em conta estes aspetos. Desta forma, as condições mais utilizadas para terminar são os seguintes [Eiben e Smith, 2003]:

- através de um tempo de processamento;
- através do número máximo de gerações;
- um valor mínimo para o desvio padrão do valor de aptidão das soluções da população;

- caso não se verificarem melhorias consideráveis das soluções durante um número de gerações consecutivas;
- através de um limiar mínimo de diversidade da população;
- através a qualidade de uma solução.

4. MÉTODOS DE RESOLUÇÃO PARA PROBLEMAS DE EMPACOTAMENTO

Neste capítulo é feita uma discrição sobre os métodos de resolução que surgiram para solucionar problemas de empacotamento. Muitas destas formas passam pela combinação de métodos heurísticos com outros métodos de pesquisa, procurando sempre a melhoria dos resultados obtidos. Existem vários métodos de pesquisa usados na resolução deste problema e alguns deste vão ser descritos neste capítulo.

4.1. Meta-heurísticas

As meta-heurísticas, são métodos de pesquisa num espaço de soluções e podem ser divididas em duas classes. Uma classe que incorpora os métodos que analisam uma vizinhança em cada iteração. A vizinhança é alterada de acordo com a sua estratégia e

nessa iteração é escolhido apenas um elemento dessa vizinhança. Esse tipo de pesquisa do espaço de soluções gera um caminho ou uma trajetória de soluções. Esta trajetória é obtida pela transição de uma solução para outra de acordo com os movimentos permitidos pela meta-heurística. Dentro desta classe de métodos podem-se citar a Pesquisa Tabu e o Arrefecimento Simulado. A segunda classe de meta-heurísticas ao contrário das anteriores, analisam uma população de soluções em cada iteração. A sua estratégia de pesquisa é capaz de analisar diversas regiões do espaço de soluções de cada vez. Nestes casos durante as iterações não se constrói uma trajetória única de pesquisa, uma vez que as novas soluções são obtidas através da combinação de soluções anteriores.

4.1.1. Algoritmos Genéticos

Os algoritmos genéticos encontram-se nesta segunda classe. Algumas abordagens para o PEC utilizam esta segunda classe de meta-heurísticas, tais como apresentam, [Bortfeldt, 1994], [Gehring e Bortfeldt, 1997], [Bortfeldt e Gehring, 2001] e [Gehring e Bortfeldt, 2002]. Bortfeldt em [Bortfeldt, 1994] melhorou a abordagem de [Gehring et al., 1990] utilizando um algoritmo genético. A abordagem inicial é utilizada para gerar uma solução inicial. Esta solução é composta por paredes que são guardadas sequencialmente. Em seguida os operadores genéticos são aplicados nessa sequência de soluções. Uma vez que o algoritmo genético usa a heurística base de [Gehring et al., 1990], os arranjos produzidos são paredes verticais sem intersecções. São elaborados testes num conjunto de 100 problemas gerados aleatoriamente e verifica-se um aumento da média da eficiência do empacotamento de 3,1% relativamente à abordagem inicial de [Gehring et al., 1990]. [Gehring and Bortfeldt, 1997], propõem uma abordagem especificamente direcionada para problemas fortemente heterogéneos. A abordagem consiste em dois passos: no primeiro passo são construídas colunas verticais de caixas (torres); no segundo passo, é utilizado um algoritmo genético para a colocação das torres no chão do contentor. Este segundo passo, é abordado como um problema bidimensional. Finalmente, a eficiência do algoritmo genético é valorizada com a introdução da hibridação. Mais precisamente, é fornecido ao algoritmo genético algum conhecimento adicional do problema base. São apresentados resultados de testes, tanto dos problemas de [Loh and Nee, 1992], como dos problemas de [Bischoff et al.,1995]. [Gehring and Bortfeldt, 1997] provam que com esta abordagem são obtidos bons resultados, quando comparados com os resultados de outras abordagens, tais como [Bischoff et al., 1995] e [Bischoff and Ratcliff, 1995]. Mais uma vez [Bortfeldt and

Gehring, 2001] e [Gehring and Bortfeldt, 2002], usam os algoritmos genéticos para a geração de planos com uma estrutura tipo parede.

Cada solução é gerada através de uma método de construção de paredes apresentado por [Gehring et al., 1990]. No início do procedimento a heurística base é usada para gerar planos de armazenamento completos, como indivíduos iniciais para a evolução. Para a geração de futuras gerações, são usados os operadores cruzamento e mutação. Isto implica uma transferência de paredes não modificadas para os descendentes. Uma vez que um descendente normalmente não representa uma solução completa, são então geradas paredes através da heurística base. A heurística base e os operadores genéticos, são tais que os planos de empacotamento são representados numa estrutura de dados por forma a não especificar a sequência das paredes. Esta sequência é só especificada no passo final do processo. Esta abordagem é direcionada para resolução de problemas fortemente heterogéneos. Foram testados os problemas de [Bischoff e Ratcliff, 1995]. Os resultados obtidos pelos mesmos autores relativamente à abordagem descrita em [Gehring and Bortfeldt, 1997], são claramente superiores. [Gehring e Bortfeldt, 2002], utilizam um algoritmo genético paralelo, a diferença fundamental para a abordagem de [Bortfeldt e Gehring, 2001] é que neste caso existem várias sub-populações, estando estas separadas e sujeitas a um processo de evolução independente. Os resultados obtidos com esta abordagem comparativamente com a anterior, são ligeiramente superiores em todos os problemas de teste.

4.1.2. Pesquisa Tabu

Estes autores usam também a Pesquisa Tabu em [Bortfeldt and Gehring, 1998] e [Bortfeldt et al., 2003] para a resolução de problemas fracamente heterogéneos (equivalente aos problemas de classe (R) segundo a classificação de [Dyckhoff, 1990]). [Bortfeldt e Gehring, 1998] apresentam um procedimento heurístico que gera soluções, essas soluções são representadas por uma string (conjunto de caracteres) de inteiros definindo uma estrutura de vizinhança. A essas soluções é aplicada uma pesquisa tabu. Para cada um dos sete primeiros conjuntos de problemas de teste de [Bischoff and Ratcliff, 1995], esta abordagem produz em média melhores resultados do que a abordagem com algoritmos genéticos de [Gehring and Bortfeldt, 1997].

Esta melhoria é compreensível, uma vez que esta abordagem é direcionada para a resolução de problemas fracamente heterogéneos, enquanto que a abordagem de [Gehring and Bortfeldt, 1997] é direcionada para problemas fortemente heterogéneos. [Bortfeldt et al., 2003] utilizam a mesma abordagem, mas com base num algoritmo de pesquisa tabu paralelo. A pesquisa paralela é efetuada através de instâncias diferentes, que cooperam entre si através da troca de (melhores) soluções no final da fase de pesquisa. Para os mesmos sete primeiros conjuntos de problemas de [Bischoff e Ratcliff, 1995], os resultados obtidos superam os resultados apresentados em [Bortfeldt e Gehring, 1998].

4.1.3. Arrefecimento Simulado

[Faina, 2000] foi dos únicos autores a utilizar o arrefecimento simulado no problema de empacotamento tridimensional. O autor apresenta o arrefecimento simulado como sendo a base de um algoritmo de empacotamento. As caixas são colocadas sem nenhuma ordem específica. Através do método da zona de [Lipovetskji, 1988] e depois de colocada a primeira caixa no contentor são obtidas as localizações para cada uma das outras caixas, obtendo-se desta forma uma configuração inicial. Nesta configuração inicial é efetuada uma pequena perturbação, por exemplo alterando (aleatoriamente) a posição de duas das caixas obtendo-se assim uma nova configuração. Esta nova configuração é aceite com uma probabilidade diferente de zero que decresce exponencialmente com o crescimento da função custo. O critério de aceitação é controlado por um parâmetro de controlo que vai diminuindo passo a passo e que reduz o número de transições aceites conforme o algoritmo vai progredindo. Esta abordagem dá bons posicionamentos das caixas quando estas são em número relativamente pequeno

4.2. Outros métodos de pesquisa

Encontram-se também na literatura outras abordagens ao PEC que utilizam ferramentas de investigação operacional sofisticadas. Um destes exemplo é em, [Han et al., 1989] e [Scheithauer, 1992] que misturam elementos heurísticos com programação dinâmica e onde o objetivo é maximizar o valor da carga que é possível empacotar num único contentor. Outra abordagem é a de [Ivancic et al., 1989] que é baseada em técnicas de programação inteira. O objetivo nesta abordagem é minimizar o custo total dos contentores

necessários para empacotar uma determinada carga. Neste caso o PEC é formulado como um problema da mochila múltiplo e resolvido usando um método de geração de colunas. É identificado um sub-problema relacionado com o empacotamento de um único contentor, que é identificado e resolvido utilizando um método heurístico.

Existem outras técnicas aplicadas aos PEC, que incluem grafos "and/or" – abordada por [Morabito e Arenales, 1993] e pesquisa em árvore - abordada por [Eley, 2002]. [Morabito e Arenales, 1993], apresentam três procedimentos heurísticos para o problema de empacotamento num único contentor. Cada uma das três abordagens é específica para cada problema sem restrições. Os autores definem problema sem restrições, se existir um número ilimitado de caixas de cada tipo. O objetivo é maximizar o volume das caixas empacotadas num único contentor. A primeira das três abordagens cai na categoria das abordagens em camadas. E reduz o problema a duas dimensões onde são geradas camadas horizontais que podem ser colocadas umas em cima das outras. Outra abordagem proposta é uma heurística de geração de colunas. As colunas são geradas resolvendo uma variedade de problemas de mochila unidimensionais. Estas colunas são colocadas no contentor através de um algoritmo de empacotamento bidimensional.

A terceira heurística é uma abordagem em grafo "and/or", que gera todos os padrões de corte possíveis para os dois problemas – sem e com restrições. [Eley, 2002], melhora as soluções iniciais da heurística construtiva, através da aplicação de uma pesquisa em árvore. Esta pesquisa permite a consideração de diferentes sequências de carga a par com as sequências do volume de utilização determinado. A ramificação é feita para cada tipo de caixa e respetivas orientações. Para limitar a ramificação, existe uma função avaliadora que indica os nós mais promissores. Esta função não só considera o volume de utilização mas também o potencial para o preenchimento dos espaços vazios com os restantes itens. Só os nós com melhor valor desta função avaliadora é que são expandidos, permitindo assim limitar a pesquisa em largura.

5. ALGORITMO IMPLEMENTADO

Neste capítulo é apresentado todo o trabalho de investigação e desenvolvimento que levou à criação do algoritmo genético proposto para o problema em questão. Nesta apresentação é exposta a estratégia adotada para o desenvolvimento do algoritmo, quais os resultados que foram obtidos durante o processo de desenvolvimento e as justificações ou conclusões que levaram a estes resultados.

5.1. Introdução

Com o aparecimento de várias propostas ou algoritmos para a solução deste problema, as meta-heurísticas destacam-se pela sua abrangente capacidade de aplicação. Entre estes os algoritmos genéticos são sempre considerados uma abordagem plausível, dada a sua capacidade de busca entre o espaço de possíveis soluções, mas no entanto existem alguns inconvenientes na utilização destes no problema de empacotamento de contentores.

Um exemplo destes inconvenientes verificou-se através de alguns resultados obtidos durante o processo de desenvolvimento do AG proposto, onde este apresentou uma eficiência pouco interessante, quando existia um processo aleatório para gerar a população inicial ou nos operadores de seleção, cruzamento ou mutação. Esta dificuldade é proporcional à dimensão do problema, ou seja, quando o número de objetos a carregar aumenta também aumenta o número de possíveis soluções do problema o que leva à necessidade de aumentar proporcionalmente o campo de pesquisa que o algoritmo precisaria para encontrar uma solução aceitável, esta é apenas uma das dificuldades encontradas com este algoritmo.

Existem outras dificuldades que foram surgindo como:

- Definir uma forma eficiente de gerar soluções iniciais;
- Definir regras para os operadores de seleção, cruzamento ou mutação;

Estas dificuldades são abordadas neste capítulo e fundamentadas com os resultados obtidos, é também explicada qual a solução adotada para cada uma das dificuldades encontradas.

Para o processo de desenvolvimento do AG a estratégia adotada baseia-se no processo de melhoramento da solução obtida, de forma a proporcionar uma evolução do algoritmo genético.

Inicialmente é construído um algoritmo genético para o empacotamento de um contentor com objetos (caixas) com dimensões variadas, de seguida é feito um teste ao algoritmo e através do resultado obtido é analisada as soluções apresentadas pelo algoritmo. Após esta análise é realizada uma reflexão sobre as possíveis melhorias que podem levar a um aumento da eficiência do algoritmo. A Figura 16 apresenta o ciclo de desenvolvimento do algoritmo.

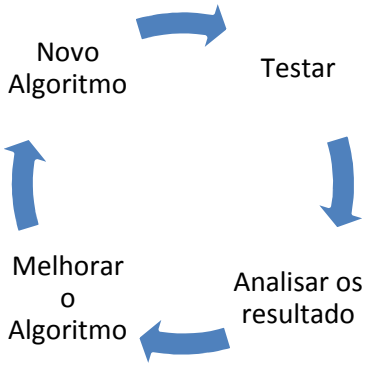


Figura 16 Ciclo de desenvolvimento

5.2. Objetivos

Pretende-se criar um *software* que contenha um AG para o empacotamento de um contentor com dimensões variáveis, onde os objetos que vão ser carregados são caixas com dimensões variáveis entre si, ou seja, cada caixa tem uma determinada dimensão. O objetivo do algoritmo é utilizar o volume disponibilizado pelo contentor de maneira mais eficiente.

Como resultado final espera-se obter uma aplicação para solucionar problemas de empacotamento em que seja, possível introduzir os dados do problema e obter da mesma uma configuração de empacotamento.

O processo de desenvolvimento passa primeiro por criar um AG e quando este estiver desenvolvido, é utilizado um componente JGAP (Java Genetic Algorithm Package) que é uma API em Java disponibilizada gratuitamente e que fornece os mecanismos básicos dos algoritmos genéticos de modo a ser possível testar o algoritmo. Os resultados obtidos através da API JGAP são representados pela API Java3D que permite a representação de objetos em 3 dimensões para facilitar a interpretação dos resultados e retirarem-se conclusões.

5.3. Formulação

O modelo matemático que serve para representação do problema tem como base o modelo apresentado por Chen, Lee em [CHEN, LEE, et al., 1995].

Supõe-se que existe apenas um contentor para carregar e que as dimensões das caixas e do contentor são conhecidas, bem como a quantidade de caixas a serem carregadas. As caixas são independentes e são posicionadas ortogonalmente no contentor, ou seja, são posicionadas em paralelo ou perpendicular aos eixos.

As caixas podem sofrer rotações de 90º no seu posicionamento, de tal forma que a largura e o comprimento da caixa não sejam posicionados obrigatoriamente em paralelo à largura e ao comprimento do contentor, respetivamente, contudo, a altura da caixa será sempre posicionada em paralelo ao eixo da altura do contentor.

Variáveis do modelo matemático:

$\boldsymbol{N}$ – número de caixas disponíveis para empacotamento;

$\boldsymbol{si}$ - variável binária que indica se a caixa foi posicionada no contentor. Quando isso ocorre, si = 1, caso contrário, si = 0;

$(\boldsymbol{L}; \boldsymbol{W}; \boldsymbol{H})$ – indica o comprimento, largura e altura do contentor, respetiva;

$(\boldsymbol{pi};\ \boldsymbol{qi};\ \boldsymbol{ri})$ - indica o comprimento, a largura e altura da caixa, respetiva;

$(\boldsymbol{xi};\ \boldsymbol{yi};\ \boldsymbol{zi})$ - indica a localização da caixa pelo canto inferior esquerdo traseiro (vértice mínimo);

$(\boldsymbol{lxi};\ \boldsymbol{lyi};\ \boldsymbol{lzi})$ - variáveis binárias que indica para qual eixo o comprimento da caixa está em paralelo. Como a altura da caixa está sempre em paralelo com a altura do contentor, então apresenta sempre o valor $(lxi; lyi; 0)$;

$(\boldsymbol{wxi};\ \boldsymbol{wyi};\ \boldsymbol{wzi})$ - variáveis binárias que indica para qual eixo a largura da caixa está em paralelo. Como a altura da caixa está sempre em paralelo com a altura do contentor, então apresenta sempre o valor $(wxi; wyi; 0)$;

$(\boldsymbol{hxi}; \boldsymbol{hyi}; \boldsymbol{hzi})$ - variáveis binárias que indica para qual eixo a altura da caixa está em paralelo. Como a altura da caixa está sempre em paralelo com a altura do contentor, então apresenta sempre o valor $(0; 0; 1)$;

Ainda existem outras variáveis que são usadas para indicar o posicionamento das caixas em relação a outras caixas:

$\boldsymbol{aik}$ - caso seja 1, indica que a caixa i está à esquerda da caixa k;

$\boldsymbol{bik}$ - caso seja 1, indica que a caixa i está à direita da caixa k;

$\boldsymbol{cik}$ - caso seja 1, indica que a caixa i está atrás da caixa k;

$\boldsymbol{dik}$ - caso seja 1, indica que a caixa i está à frente da caixa k

$\boldsymbol{eik}$ - caso seja 1, indica que a caixa i está abaixo da caixa k;

$\boldsymbol{fik}$ - caso seja 1, indica que a caixa i está acima da caixa k;

O problema então é formulado conforme abaixo, onde se observa que o objetivo é

Minimizar o espaço não preenchido do contentor:

$$min\ L.W.H - \sum_{i=1}^{N} pi.qi.ri.si$$

Sujeito a

(Evitar sobreposições de caixas no contentor)

$$x_i + p_i.l_{xi} + q_i.w_{xi} \leq x_k + (1 - c_{ik}).M, \forall_i, k, i < k \qquad (1)$$

$$x_k + p_k.l_{xk} + q_k.w_{xk} \leq x_i + (1 - d_{ik}).M, \forall_i, k, i < k \qquad (2)$$

$$y_i + p_i.l_{yi} + q_i.w_{yi} \leq y_k + (1 - a_{ik}).M, \forall_i, k, i < k \qquad (3)$$

$$y_k + p_k.l_{yk} + q_k.w_{yk} \leq y_i + (1 - b_{ik}).M, \forall_i, k, i < k \qquad (4)$$

$$z_i + p_i.l_{zi} + q_i.w_{zi} + r_i.h_{zi} \leq z_k + (1 - e_{ik}).M, \forall_i, k, i < k \quad (5)$$

$$z_k + p_k.l_{zk} + q_k.w_{zk} + r_k.h_{zk} \leq y_i + (1 - b_{ik}).M, \forall_i, k, i < k \quad (6)$$

(Garantir que o par de caixas avaliados nas equações (1) a (6) está no contentor)

$$a_{ik} + b_{ik} + c_{ik} + d_{ik}.+e_{ik} + f_{ik} \geq s_i + s_k - 1, \forall_i, k, i < k \quad (7)$$

(Garantir que o posicionamento das caixas obedece as limitações físicas dimensionais do contentor)

$$x_i + p_i.l_{xi} + q_i.w_{xi} \leq L + (1 - s_i).M, \forall_i \quad (8)$$

$$y_i + p_i.l_{yi} + q_i.w_{yi} \leq W + (1 - s_i).M, \forall_i \quad (9)$$

$$z_i + p_i.l_{zi} + q_i.w_{zi} + r_i.h_{zi} \leq H + (1 - s_i).M, \forall_i \quad (10)$$

M é um número inteiro arbitrário e grande.

As variáveis binárias que identificam o posicionamento das caixas com relação aos eixos são variáveis dependentes e relacionam-se conforme abaixo:

$$l_{xi} + l_{yi} + l_{zi} \leq 1 \quad (11)$$

$$w_{xi} + w_{yi} + w_{zi} \leq 1 \quad (12)$$

$$z_{xi} + z_{yi} + z_{zi} \leq 1 \quad (13)$$

$$l_{xi} + w_{yi} + z_{zi} \leq 1 \quad (14)$$

$$l_{yi} + w_{yi} + z_{yi} \leq 1 \quad (15)$$

$$l_{zi} + w_{zi} + z_{zi} \leq 1 \quad (16)$$

5.4. Algoritmo Genético

Gene

O gene representa uma caixa do problema a empacotar e tem a seguinte informação.

Load	XPointMin	YPointMin	ZPointMin	Posição

Load - variável binária que indica se a caixa foi posicionada no contentor. Quando isso ocorre é 1, caso contrário é 0;

XPointMin – variável inteira que indica a coordenada X da caixa pelo canto inferior esquerdo traseiro

YPointMin - variável inteira que indica a coordenada Y da caixa pelo canto inferior esquerdo traseiro

ZPointMin - variável inteira que indica a coordenada Z da caixa pelo canto inferior esquerdo traseiro

Posição – variável que indica qual o posicionamento da caixa.

Se valor igual 1 então o comprimento da caixa está em paralelo com o comprimento do contentor $(\boldsymbol{lxi}; \boldsymbol{lyi}; \boldsymbol{lzi})$ é representado por $(1; 0; 0)$ e $(\boldsymbol{wxi}; \boldsymbol{wyi}; \boldsymbol{wzi})$ é representado por $(0; 1; 0)$;

Se valor igual a 2 comprimento da caixa é perpendicular ao comprimento do contentor $(\boldsymbol{lxi}; \boldsymbol{lyi}; \boldsymbol{lzi})$ é representado por $(1; 0; 0)$ e $(\boldsymbol{wxi}; \boldsymbol{wyi}; \boldsymbol{wzi})$ é representado por $(0; 1; 0)$.

Cromossoma

É formado pelo conjunto de genes onde cada gene representa uma caixa. O cromossoma é composto por todas as caixas que existem para serem carregadas.

Gene1	Gene1	…	GeneN

Função Aptidão

A avaliação de uma possível solução é feita pela sua classificação de 0 a 100. Esta classificação é calculada através de dois métodos, primeiro pela verificação do volume utilizado e segundo pelo número de restrições que são cumpridas.

Verificação do volume utilizado

Neste método é calculada a percentagem de volume que as caixas empacotadas utilizam no volume do contentor:

$$\frac{\sum_{i=1}^{N} pi.qi.ri.si}{L.W.H} * 100$$

Caso se verifique que as caixas a empacotar já estão todas empacotadas na solução então é atribuída uma percentagem de utilização máxima (100%).

Verificação do número de restrições

Depois de atribuída uma classificação segundo o volume do contentor utilizado pelas caixas é feita uma verificação do cumprimento das restrições do problema.

Verifica-se então se existe alguma restrição que não é cumpridas pela solução apresentada e por cada restrição e é feita uma penalização de 5% à classificação atribuída pelo método da verificação do volume utilizado.

5.5. *Software*

A aplicação desenhada para implementar o algoritmo tem como função, ler os dados do problema e representar graficamente a solução apresentada pelo AG para o problema.

A Figura 17 apresenta o *layout* da janela principal da aplicação. Pode-se observar que a janela é composta por um painel, uma caixa de texto e vários botões. Os componentes

apresentados na Figura 17 pertence à *framework Swing* que fornece vários componentes escritos em java para construir interfaces gráficas.

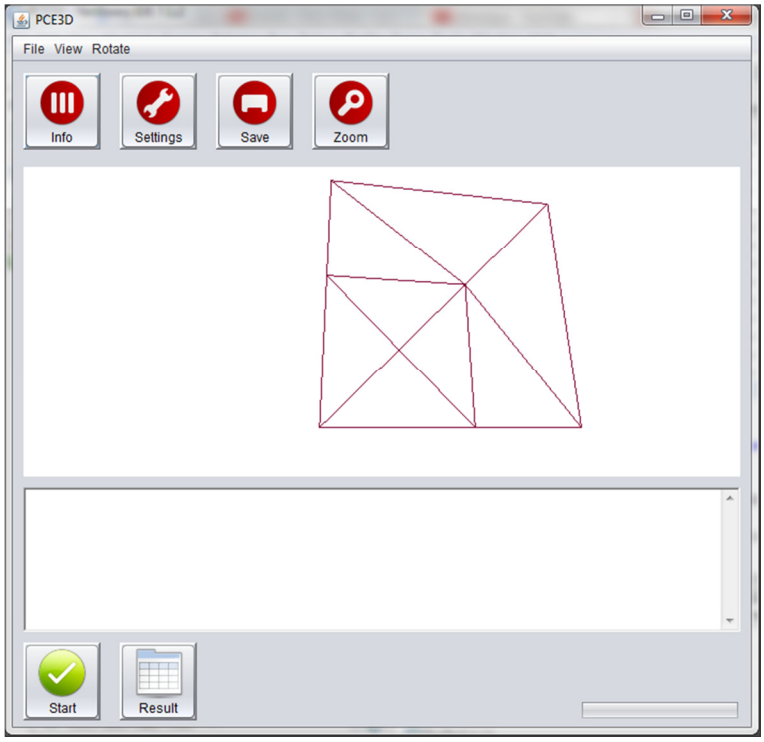


Figura 17 Janela Principal

O painel serve para apresentar as soluções graficamente proposta pelo AG, para auxiliar esta representação é utilizada a API Java 3D que será abordada nos próximos subcapítulos.

A caixa de texto tem como principal função apresentar toda a informação sobre a solução do problema de empacotamento, como o tempo de cálculo, número total de caixas a carregar, volume ocupado e posição de empacotamento de cada caixa.

Os vários botões que são vistos na Figura 17 têm diversas funções como aceder outras janelas ou começar o algoritmo.

Info: Aceder a janela de configuração das caixas e contentores;

Settings: Aceder à janela de configuração do AG (tamanho da população inicial, numero de evoluções);

Save: Guardar a solução num ficheiro;

Zoom: Aumentar o tamanho do painel de representação gráfica;

Start: Iniciar o AG;

Result: Mostrar os dados dos genes do cromossoma final;

A janela de configuração do problema de empacotamento apresenta na Figura 18 é composta por duas tabelas onde é introduzido as características do contentor e das várias caixas que se pretende empacotar no contentor como as dimensões respetivas.

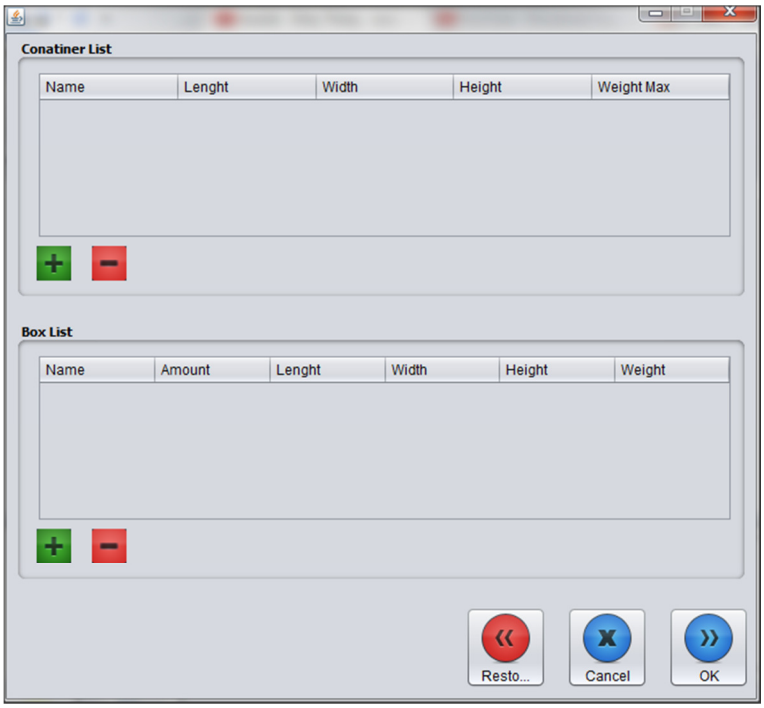


Figura 18 Janela de configuração do Problema

A janela de configuração do AG ilustrada na Figura 19 é composta por caixas de texto que servem para introduzir o tamanho da população inicial e o número de evoluções que o AG vai realizar.

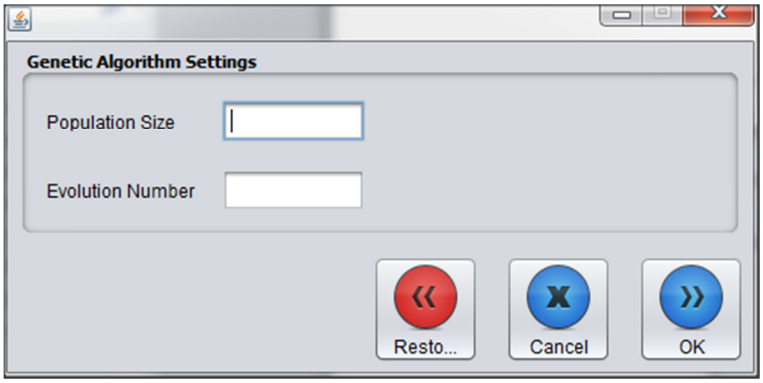


Figura 19 Janela de configuração AG

As janelas referidas compõem a aplicação onde é implementado o AG. A linguagem utilizada é java, como as API utilizadas para auxiliar a construção desta aplicação.

5.5.1. JGAP

Para auxiliar a criação de AG recorreu-se a API JGAP que consiste num pacote de classes Java de uso livre e que fornecem os mecanismos básicos, que podem ser usados para aplicar em princípios evolutivos para soluções de problemas.

Para utilizar este componente é necessário definir o cromossoma, a função aptidão e todo o mecanismo de evolução é feito pelo componente. Nesta situação a população inicial e os operadores de evolução recorrem a processos aleatórios. Este componente também possibilita a configuração dos operadores de evolução. Quando o processo de evolução é terminado o componente devolve a solução com a melhor qualificação atribuída pela função aptidão.

5.5.2. JAVA3D

A API Java3D é em uma hierarquia de classes Java que servem como interface para o desenvolvimento de sistemas gráficos tridimensionais. É composta por construtores de alto nível que permitem a criação e manipulação de objetos geométricos, especificados num universo virtual. Também possibilita a criação de universos virtuais com uma grande flexibilidade, onde as cenas são representadas através de grafos e os detalhes da sua visualização são gerados automaticamente. Assim o desenvolvimento de um programa em Java3D resume-se a criação de objetos e do seu posicionamento num grafo de cena, que os

combina numa estrutura em árvore. Os grafos de cena são responsáveis pela especificação do conteúdo do universo virtual e pela forma como este é visualizado.

5.6. Implementação do Algoritmo Genético

5.6.1. 1º Implementação

Neste primeiro ensaio o AG vai ter o gerador de população inicial e os operadores de evolução a operar aleatoriamente comportando-se como mostra Figura 20.

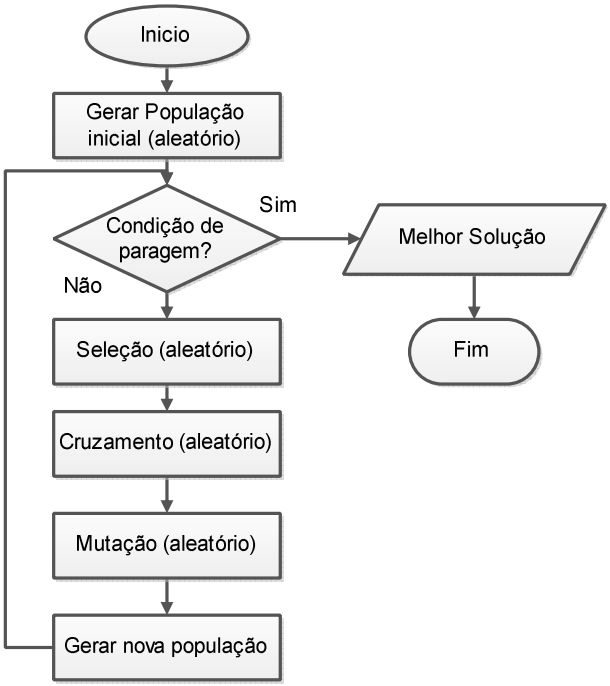


Figura 20 AG primeiro ensaio

5.6.2. Configuração do problema

Para iniciar os testes do AG considerou-se o seguinte problema de empacotamento:

- 1 Contentor de dimensões: 5 comprimento, 5 largura, altura.
- 4 Caixas de dimensões: 1 comprimento, 1 largura, 1 altura.

A Figura 21 mostra a aplicação com a configuração do problema.

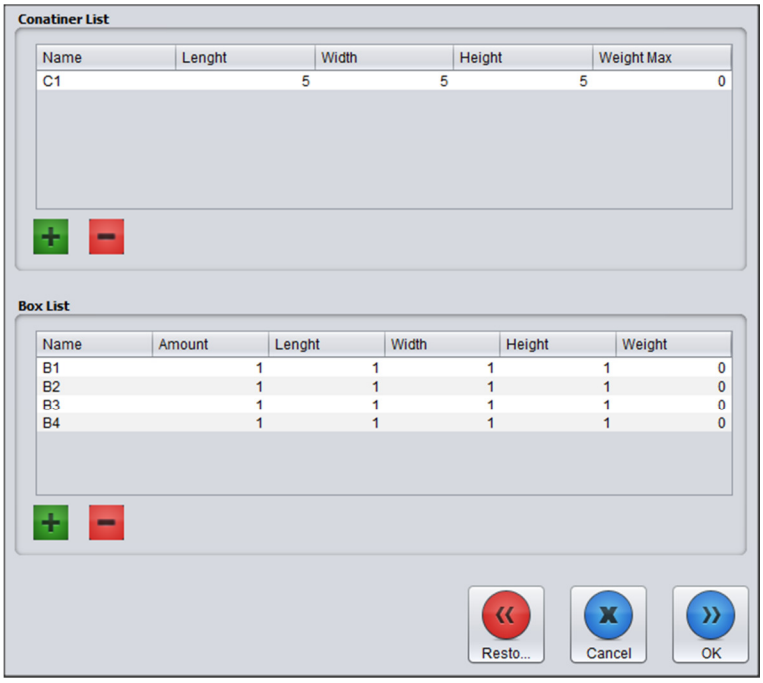


Figura 21 Configuração do Problema

5.6.2.1. 1º Teste

As definições de evolução e do tamanho da população do AG são as indicadas na Figura 22.

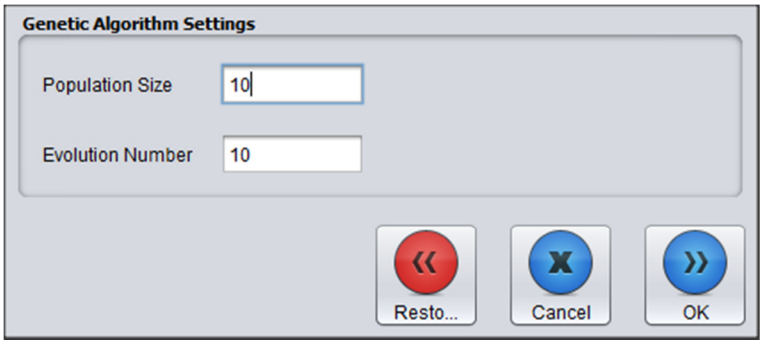


Figura 22 Configuração do AG teste 1

O resultado do 1º teste é mostrado na Figura 23.

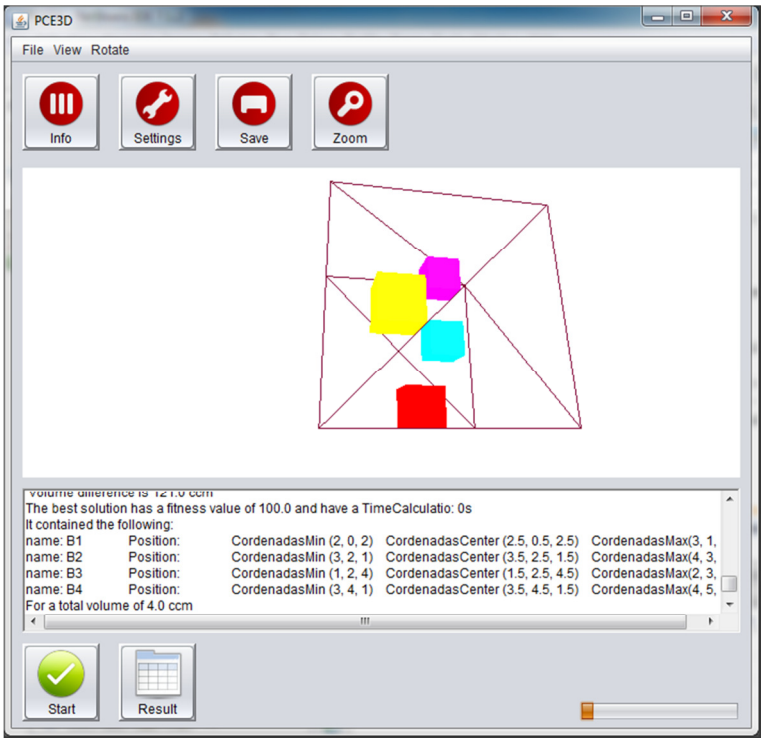


Figura 23 Teste1- Resultado

A Figura 24 apresenta a solução final do 1º teste em vários ângulos.

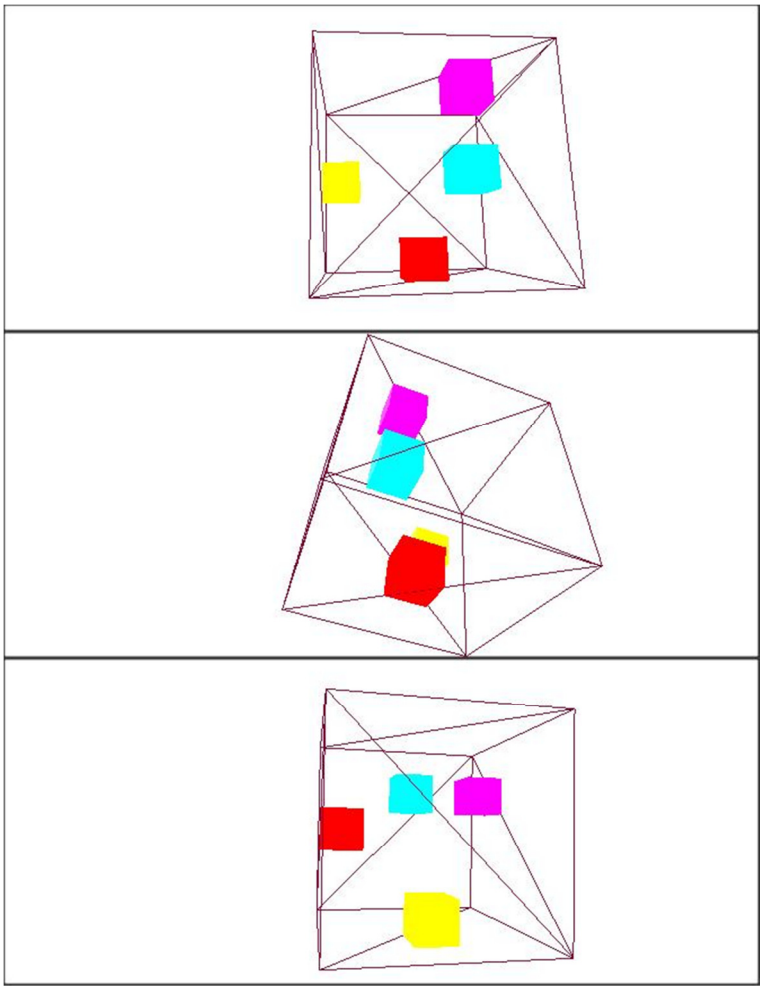


Figura 24 Teste1- Varias Vistas

Características da população em função dos genes de cada indivíduo.

----cromossoma 0-----
Load: true Xaxis: 4 Yaxis: 3 Zaxis: 3 PositionGene: 2
Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1
Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 2
Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2
----cromossoma 1-----
Load: true Xaxis: 4 Yaxis: 3 Zaxis: 3 PositionGene: 2
Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 2-----

Load: true Xaxis: 4 Yaxis: 3 Zaxis: 3 PositionGene: 2

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 3-----

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 4 Zaxis: 1 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 4-----

Load: true Xaxis: 4 Yaxis: 3 Zaxis: 3 PositionGene: 2

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 5-----

Load: true Xaxis: 4 Yaxis: 3 Zaxis: 3 PositionGene: 2

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 6-----

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 4 Zaxis: 1 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 7-----

Load: true Xaxis: 1 Yaxis: 3 Zaxis: 1 PositionGene: 2

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 0 Zaxis: 4 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 8-----

Load: true Xaxis: 4 Yaxis: 3 Zaxis: 3 PositionGene: 2

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 4 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

----cromossoma 9-----

Load: true Xaxis: 4 Yaxis: 3 Zaxis: 3 PositionGene: 2

Load: true Xaxis: 1 Yaxis: 1 Zaxis: 1 PositionGene: 1

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 4 PositionGene: 2

Load: true Xaxis: 3 Yaxis: 1 Zaxis: 0 PositionGene: 2

5.6.2.2. 2º Teste

Alterar as definições de evolução e do tamanho população do AG como mostra a Figura 25.

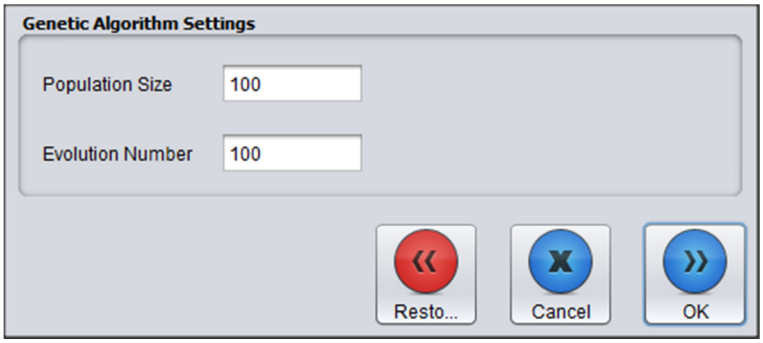


Figura 25 Configuração AG Teste 2

O resultado do 2º teste é mostrado na Figura 26.

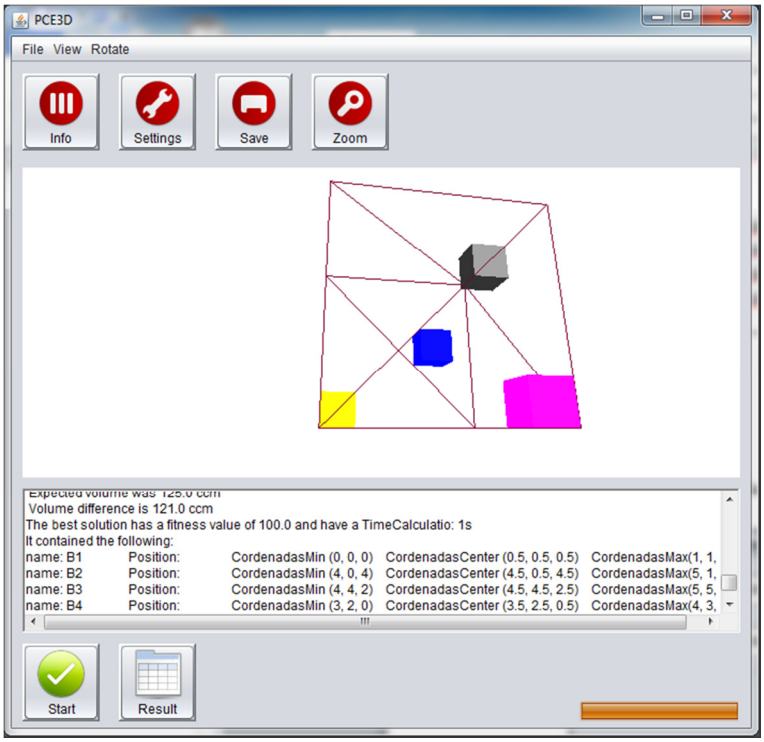


Figura 26 Teste2-Resultado

5.6.3. 3º Teste

Alterar o número de caixas para 50 com dimensões 1*1*1 como mostra a Figura 27.

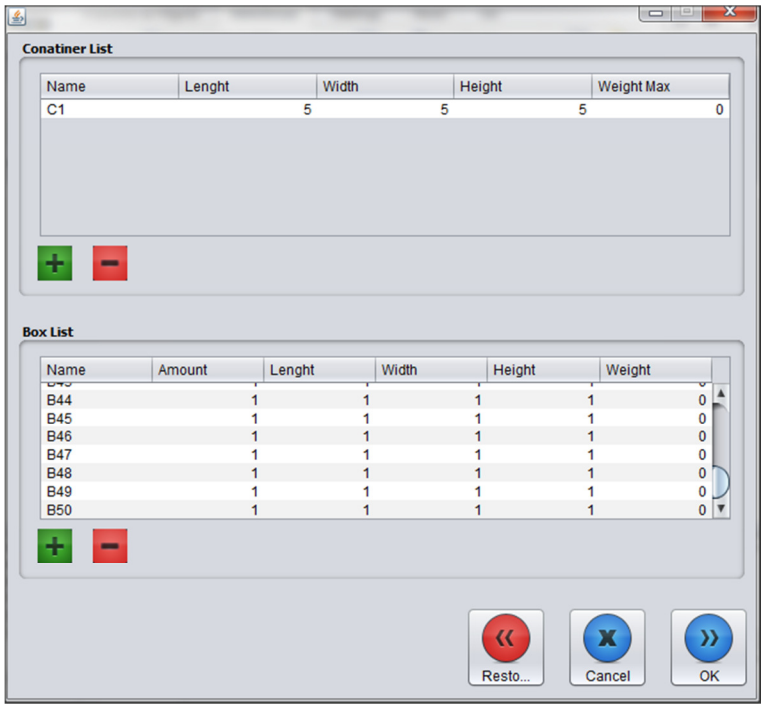


Figura 27 Configuração do Problema Teste 3

O resultado obtido com o 3º teste é mostrado na Figura 28.

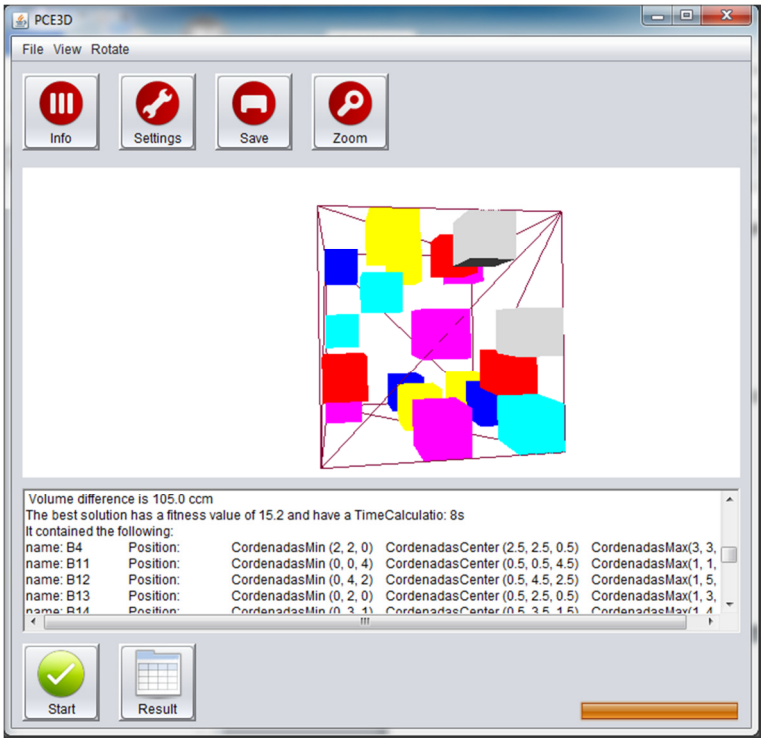


Figura 28 Teste3-Resultado 1

A Figura 29 apresenta a solução final do 3º teste em vários ângulos.

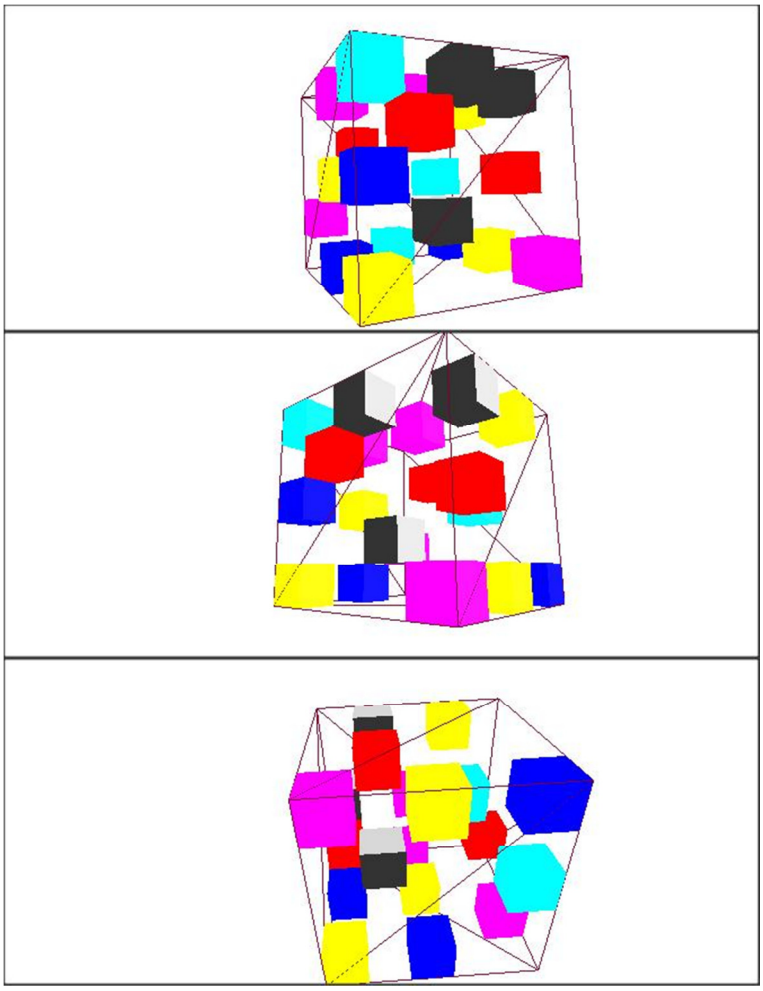


Figura 29 Teste 3-Resultado 2

Conclusões

Nestes três testes verificou-se que o algoritmo, apresentou soluções que respeitam as restrições de empacotamento e seleciona todas as caixas para empacotar, mas estas soluções são inviáveis porque a maior parte das vezes as caixas encontram-se a flutuar. Este facto torna este algoritmo ineficiente para o problema em questão.

Possíveis Melhorias

Com os resultados obtidos na primeira implementação do algoritmo genético observou-se que uma forma de obter uma melhor performance do GA é garantir que a população inicial não contém soluções inviáveis e repetidas de forma a otimizar o tempo de busca.

5.6.4. 2º Algoritmo Genético

Para criar uma população inicial otimizada recorreu-se a uma heurística que garanta que a população inicial contém apenas soluções viáveis.

Existem vários heurística que podem fazer esta tarefa, a escolhido para este trabalho é a "Heurística de Suavização de Superfícies Irregulares" (HSSI), apresentado por [Aloise, Gonzaga e Bittencourt, 2003], que tenta resolver problemas de empacotamento simulando o comportamento da maioria das pessoas ao fazer este processo na vida real. No quotidiano revela que a maioria das pessoas, ao efetuarem um empacotamento, onde existe vários objetos diferentes, tem a tendência de escolher sempre as caixas cujas dimensões se aproximem o máximo possível do espaço disponível para o empacotamento. No caso de ainda existir espaço desocupado, a próxima caixa a ser escolhida é aquela que preencha o tal espaço de modo a que o topo de ambas as caixas forme uma superfície sem descontinuidade, ou seja, "suavizada". A HSSI reflete o mesmo comportamento.

A HSSI apresenta o seguinte funcionamento:

1. Verificar as caixas a carregar.
2. Criar uma camada caso exista espaço no contentor
    2.1.1. Escolher a melhor caixa para preencher a camada;
    2.1.2. Enquanto houver espaço na camada, escolher a melhor caixa de forma a obter uma superfície plana com as caixas, caso contrário voltar para 2;
3. Quando não existir caixas a empacotar ou espaço disponível no contentor termina

Perante um número de caixas a serem empacotadas, verifica-se as dimensões dos espaços disponíveis, as caixas que podem preencher esse espaço, considerando todas as suas respetivas restrições de posicionamento, e procura-se a melhor caixa de modo a garantir uma suavização da superfície.

Na literatura esta forma de criação de padrões ainda não foi experimentada com algoritmos genéticos.

Neste segundo ensaio o AG vai ter um gerador da população inicial controlado pela HSSI e sem operadores de evolução como mostra Figura 30.

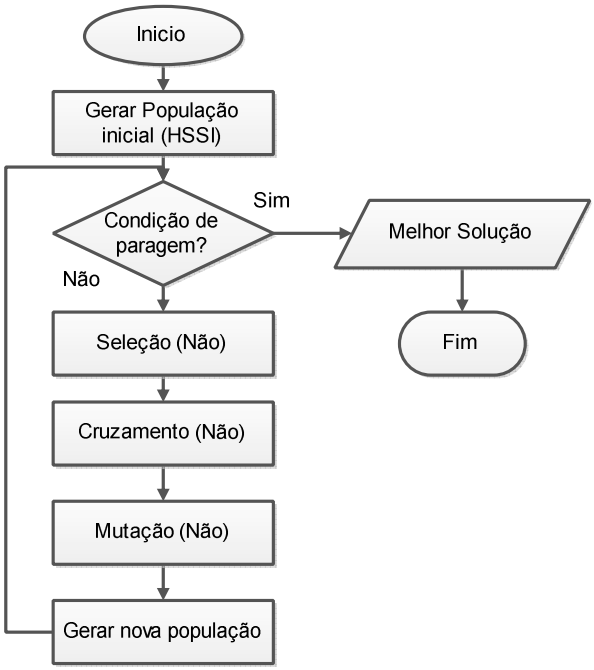


Figura 30 AG Segundo ensaio

Comportamento da HSSI

Neste teste é feita uma exemplificação de como a HSSI cria e preenche as camadas. Existem 6 caixas de dimensões variadas para carregar como mostra a Figura 31.

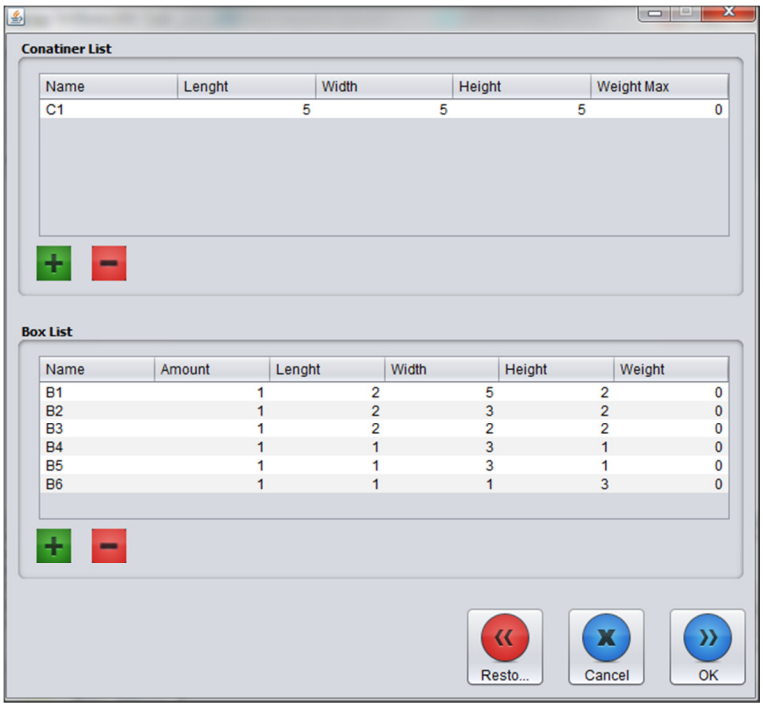


Figura 31 Configuração do problema (HSSI)

1º Camada

Para escolher o tamanho da camada o algoritmo baseia-se nas dimensões da caixa com maior volume, ou seja, vai analisar qual a caixa que tem maior volume entre as caixas a carregar e cria uma camada com as mesmas dimensões se esta respeitar o tamanho máximo do contentor.

Neste problema a caixa com maior volume é a "B1" então a HSSI vai criar uma camada de tamanho (5,2,2). As dimensões da camada não correspondem totalmente ao tamanho da caixa, porque houve uma rotação do eixo x com o eixo z que é um critério da HSSI, que aplica o maior tamanho entre a largura e o comprimento da caixa ao eixo do x da camada como mostra a Figura 32. Esta camada vai ser posicionada no menor canto inferior esquerdo. Quando definido o tamanho e a posição da camada, segue-se então para a busca de uma caixa que preencha melhor a camada definida como se pode ver na Figura 32. Esta sequência de processo não parece muito logica na criação da 1ª camada mas torna-se útil nas próximas.

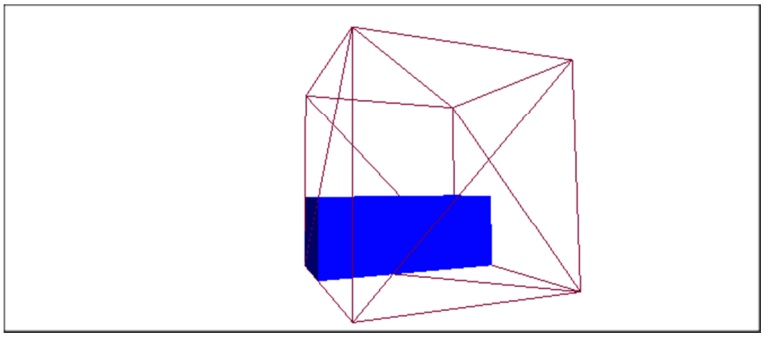


Figura 32 HSSI - 1ª Camada

Tamanho (x,y,z)	Posição (Vértice traseiro inferior esquerdo) (x,y,z)
(5,2,2)	(0,0,0)

2ª Camada

Para criar a camada seguinte o algoritmo vai ter em conta dois critérios.

O 1º critério é analisar se existem caixas que ainda não foram carregadas.

O 2º critério é verificar se existe a possibilidade de criar uma camada em cima da anterior de forma a maximizar a altura utilizada. Esta nova camada, que será criada em cima da anterior, terá que ter como tamanho máximo a largura e o comprimento da anterior, a altura será o remanescente entre a camada anterior e altura do contentor. Este critério leva o algoritmo a construir colunas de camadas.

Caso não seja possível criar uma camada em cima da anterior o algoritmo tenta criar uma camada a frente da anterior, avançando portanto, a posição da camada no eixo z como veremos mais a frente.

Para este problema a 2ª camada é preenchida como mostra a Figura 33.

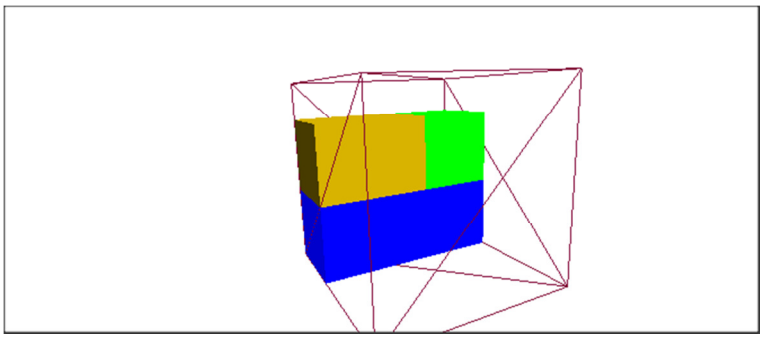


Figura 33 HSSI - 2ª Camada

Tamanho (x,y,z)	Posição (Vértice traseiro inferior esquerdo) (x,y,z)
(5,2,2)	(0,2,0)

Como mostra a Figura 33 a camada foi criada em cima da anterior e as dimensões máximas de cumprimento e largura obedecem a dimensões máximas da camada anterior.

Neste caso verifica-se que o algoritmo utilizou duas caixas para maximizar a utilização do espaço que a camada dispõe.

Este critério só acontece quando as caixas são da mesma altura como mostra a Figura 33.

3ª Camada

Esta próxima camada vai ser criada e preenchida seguindo os mesmos critérios que a camada anterior e o resultado será o mostrado a Figura 34.

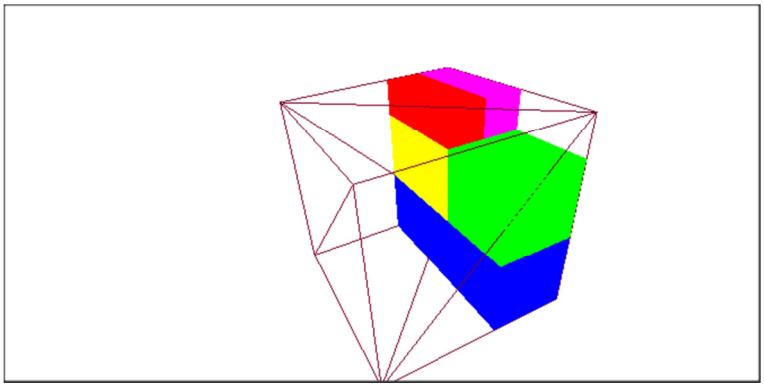


Figura 34 HSSI - 3ª Camada

Tamanho (x,y,z)	Posição (Vértice traseiro inferior esquerdo) (x,y,z)
(5,1,2)	(0,4,0)

Nesta cama foram utilizadas duas caixas para maximizar o espaço.

4ª Camada

Seguindo os mesmos critérios que a camada anterior e o resultado será como mostra a Figura 35.

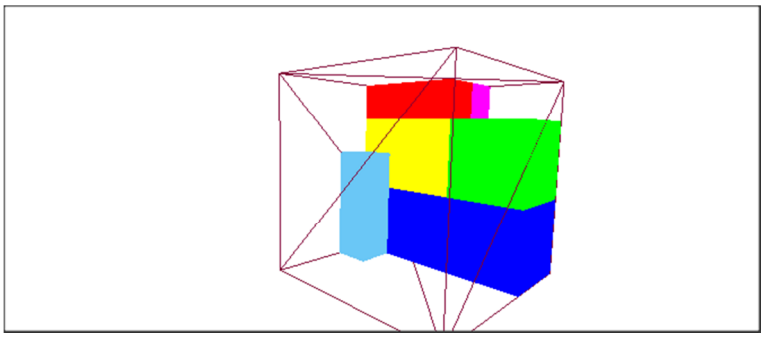


Figura 35 HSSI - 4ª Camada

Tamanho (x,y,z)	Posição (Vértice traseiro inferior esquerdo) (x,y,z)
(1,3,1)	(0,0,2)

Nesta camada verifica-se que o algoritmo criou uma camada não acima da anterior mas a frente.

Este processo de criação e preenchimento de camadas repete-se até o contentor estiver cheio, ou as caixas estiverem todas carregadas ou não existir espaços livres com dimensões para a alocar novas caixas.

5.6.5. 4º Teste

Neste teste é verificado o comportamento da HSSI alterando o número de caixas para 50 com dimensões 1*1*1 como mostra a Figura 36.

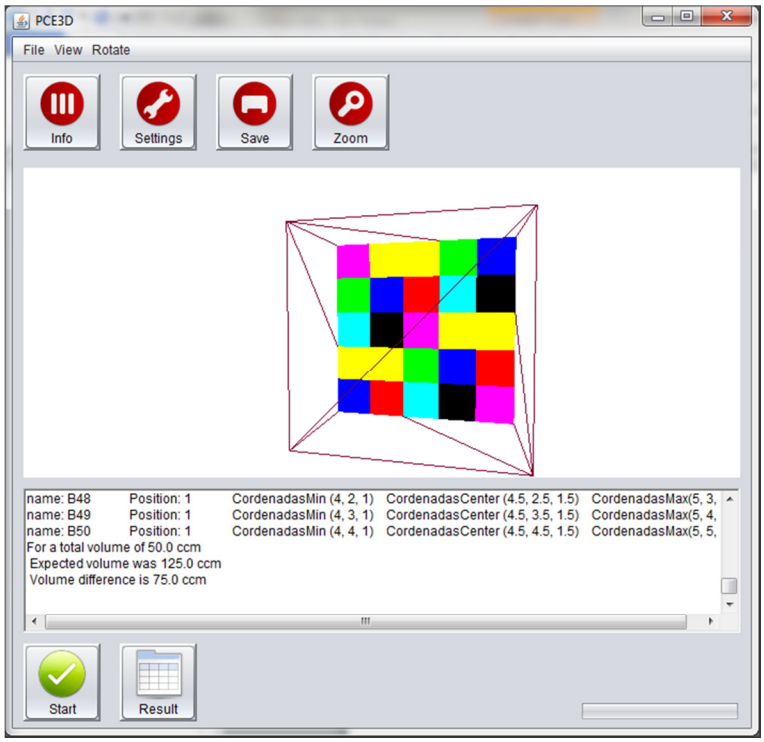


Figura 36 Teste 4-Resultado 2

A figura 37 apresenta a solução final em vários ângulos do 4º teste.

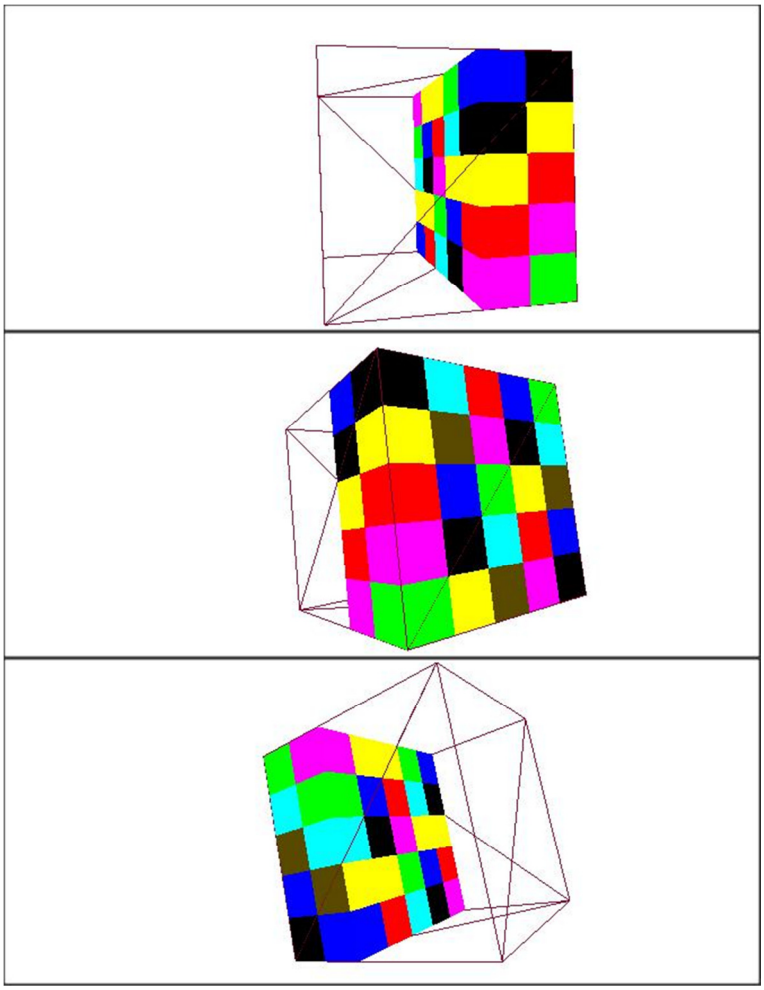


Figura 37 Teste 4-Resultado 3

5.6.6. 5º Teste

Neste teste é alterado o numero de caixa a empacotar o e as dimensões das caixas como mostra a tabela 3.

Tabela 3 Caixas a Empacotar 5º teste

Nome	Quantidade	Comprimento	Largura	Altura	Peso
B1	1	2	1	1	0.0
B2	1	2	2	2	0.0
B3	1	2	3	1	0.0

B4	1	2	1	2	0.0
B5	1	2	4	1	0.0
B6	1	2	2	2	0.0
B7	1	2	3	1	0.0
B8	1	2	2	3	0.0
B9	1	2	4	1	0.0
B10	1	1	1	1	0.0
B11	1	1	1	2	0.0
B12	1	1	1	1	0.0
B13	1	1	1	2	0.0
B14	1	1	1	1	0.0
B15	1	1	1	4	0.0
B16	1	1	1	5	0.0
B17	1	1	1	2	0.0
B18	1	1	1	1	0.0
B19	1	1	1	2	0.0
B20	1	1	1	1	0.0
B21	1	1	1	1	0.0
B22	1	1	1	1	0.0
B23	1	2	1	1	0.0
B24	1	2	2	2	0.0
B25	1	2	3	1	0.0

Aplicado o AG o resultado do 5º teste é o mostrado na Figura 38.

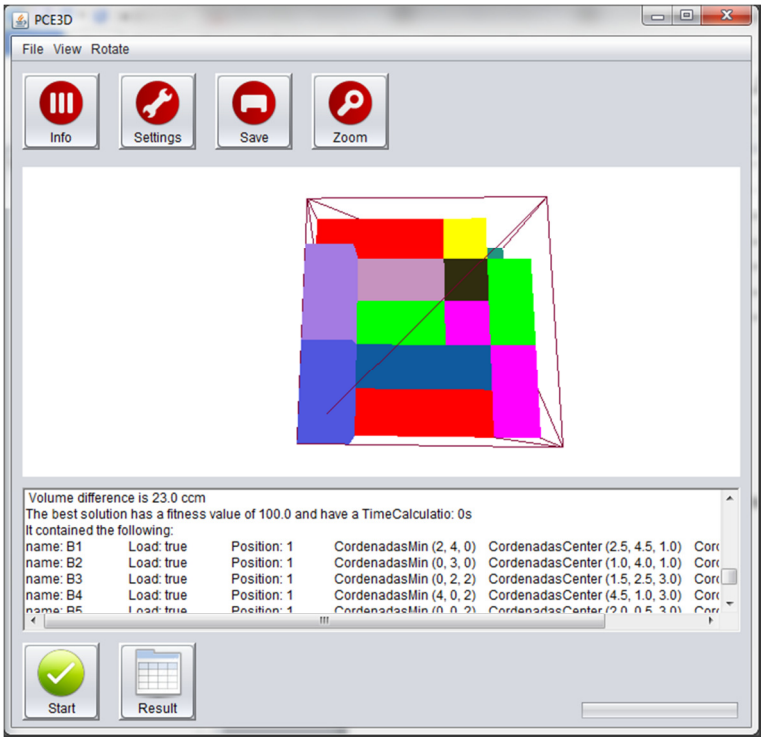


Figura 38 Teste 5-Resultado 1

A figura 39 apresenta a solução final do 5º teste em vários ângulos.

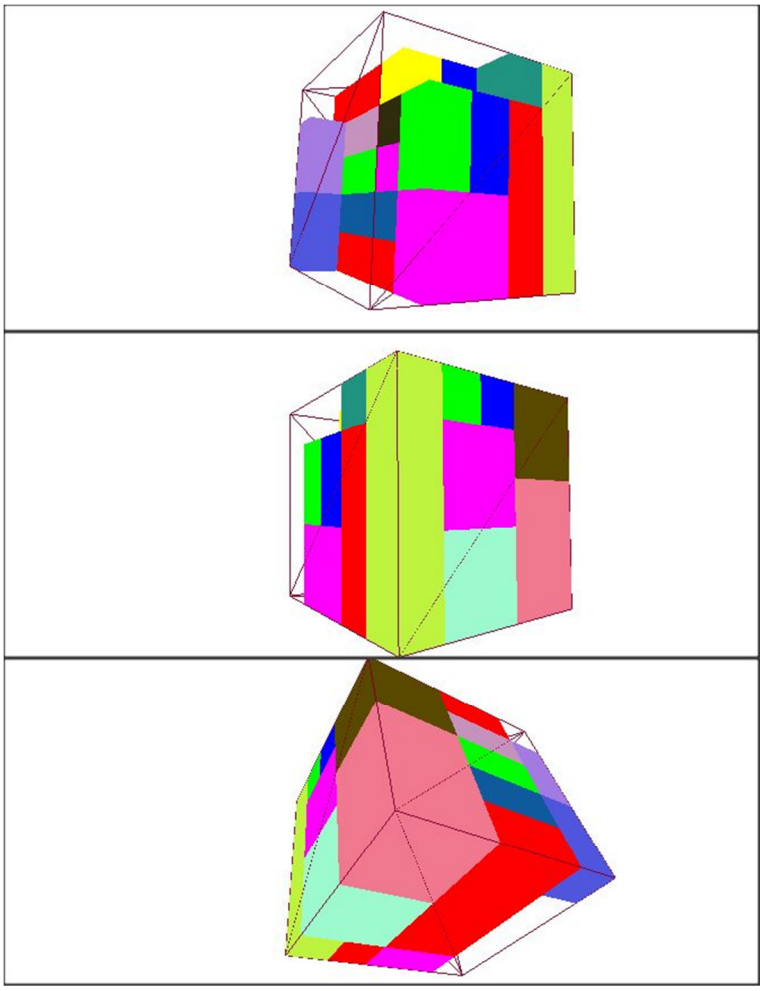


Figura 39 Teste 5-Resultado 2

Fator aleatório na HSSI

Nos testes anteriores verificou-se que heurística HSSI pode apresentar bons resultados, mas como gerador de uma população inicial não é muito interessante dado que apresenta sempre a mesma solução para um dado problema, urge então a necessidade de criar um fator aleatório para que seja possível que esta heurística apresente varias soluções possíveis e diferentes entre si para o mesmo problema.

Uma proposta para diversificar as soluções iniciais é alternar a caixa que serve como suporte para criar a primeira camada, ou seja, utilizar um método dinâmico para definir as dimensões da primeira camada.

A escolha das caixas é iniciada pela caixa de maior volume e termina na caixa de menor volume, quando terminada esta sequência repete-se mas troca-se o posicionamento das caixas trocando a largura com o cumprimento.

5.6.7. 6º Teste

Neste teste é alterado o numero de caixa a empacotar o e as dimensões das caixas como mostra a tabela 5.

Tabela 4 Caixas a Empacotar 6º teste

Nome	Quantidade	Comprimento	Largura	Altura	Peso
B1	1	5	5	2	0.0
B2	1	5	2	2	0.0
B3	1	5	2	2	0.0
B4	1	2	3	1	0.0
B5	1	2	3	1	0.0
B6	1	1	1	3	0.0
B7	1	1	2	3	0.0

O resultado obtido com o 6º teste é mostrado na Figura 40.

5 Caixas empacotadas

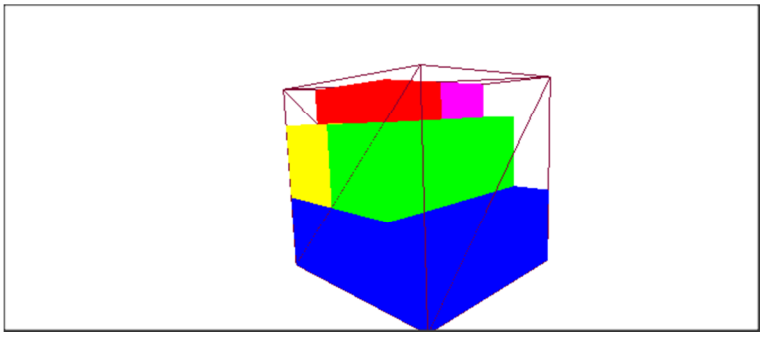


Figura 40 Teste 6-Resultado

Neste teste observa-se que são empacotadas 5 caixas num total de 7. As caixas que não foram alocadas são as caixas B6 e B7 segundo os resultados obtidos pelo cromossoma final.

----cromossoma 0-----
Load: true Xaxis: 0 Yaxis: 0 Zaxis: 0 PositionGene: 1
Load: true Xaxis: 0 Yaxis: 2 Zaxis: 0 PositionGene: 1
Load: true Xaxis: 2 Yaxis: 2 Zaxis: 0 PositionGene: 1
Load: true Xaxis: 0 Yaxis: 4 Zaxis: 0 PositionGene: 1
Load: true Xaxis: 0 Yaxis: 4 Zaxis: 2 PositionGene: 1
Load: false Xaxis: 0 Yaxis: 0 Zaxis: 0 PositionGene: 1
Load: false Xaxis: 0 Yaxis: 0 Zaxis: 0 PositionGene: 1

Observa-se que o espaço não ocupado é suficiente para alocar as caixas, que não foram alocadas na solução final, o que leva a concluir que o padrão de alocação de caixas, em camadas, apresenta uma limitação. Esta limitação prende-se com o facto de o algoritmo,

ver apenas o espaço livre de cada a camada e não resultado dos espaços livres, ou seja, a união do espaço livre de todas as camadas.

Esta limitação como se verificou neste teste, pode influenciar de forma negativa a solução final, de cada problema dado esta situação pode facilmente acontecer.

Conclusões

Nestes quatro testes verificou-se que o algoritmo para gerar a população inicial, apresentou soluções interessantes que respeitam as restrições de empacotamento e também alguma racionalização do espaço do contentor.

Contudo verificasse que existe falhas no aproveitamento do espaço, ou seja, podia haver uma melhor utilização do espaço do contentor como se viu no 6º teste.

Possíveis Melhorias

Uma melhoria evidente será a de dar a capacidade de o algoritmo saber o espaço livre que não foi ocupado pela HSSI.

Esta capacidade pode ser introduzir nos fatores de evolução de modo a alocar as caixas que não estão carregadas nos espaços remanescentes.

5.6.8. 3º Algoritmo Genético

Para diversificar o campo de procura foi introduzido no GA anterior um operador genético de mutação. Este operador vai tentar completar a alocação proposta pela HSSI que como se verificou apresenta falhas na alocação das caixas. Estas falhas acontecem devido ao facto de o HSSI trabalhar camada a camada e não saber qual o espaço livre resultante de todas as camadas.

O operador genético que é introduzido tem como tarefa unir espaços vazios das camadas caso seja possível e alocar uma caixa a esse novo espaço, que ainda não conste na configuração de empacotamento. Contudo este operador não vai operar caso se verifique que já não existe caixas a empacotar.

Neste ensaio o AG vai ter um gerador da população inicial controlado pela HSSI e com um operador de mutação como mostra Figura 41.

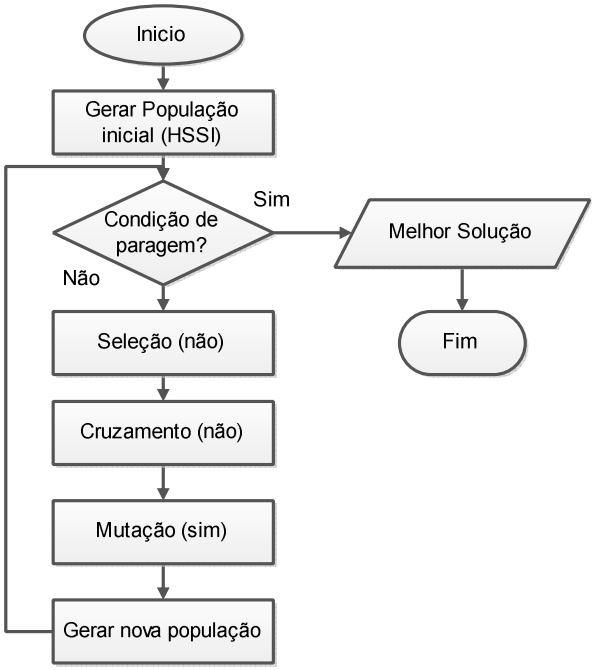


Figura 41 AG Terceiro ensaio

5.6.9. 7º Teste

Para este teste é utilizado a mesma configuração do problema do 6º como mostra a tabela 5.

Tabela 5 Caixas a Empacotar

Nome	Quantidade	Comprimento	Largura	Altura	Peso
B1	1	5	5	2	0.0
B2	1	5	2	2	0.0
B3	1	5	2	2	0.0
B4	1	2	3	1	0.0
B5	1	2	3	1	0.0
B6	1	1	1	3	0.0
B7	1	1	2	3	0.0

Os resultados obtidos com esta nova versão do algoritmo são apresentados na Figura 42.

7 Caixas empacotadas

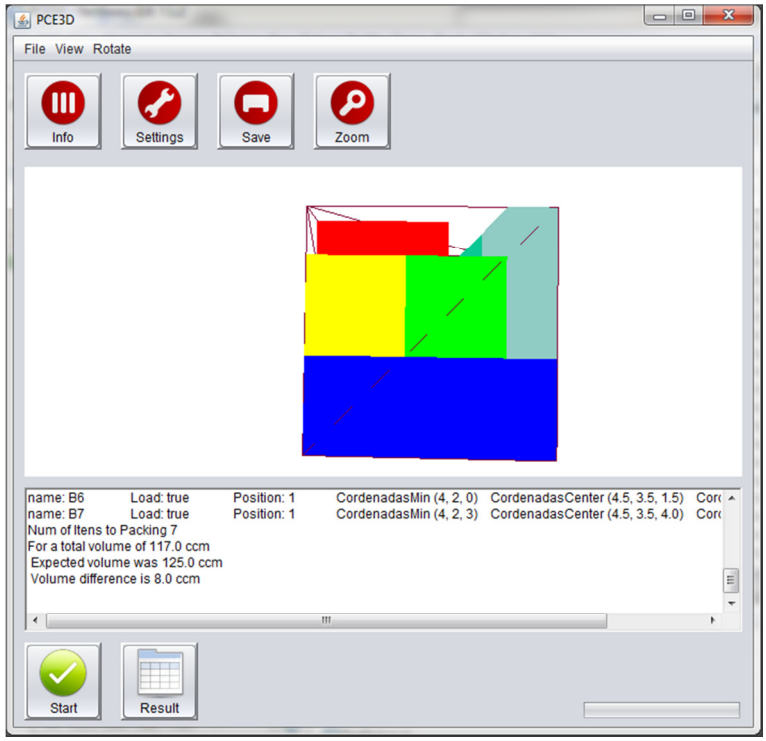


Figura 42 Teste 7-Resultado

Com esta nova função do algoritmo observa-se, que as caixas são todas empacotadas no mesmo espaço, ou seja, com este operador mutação é garantido que o espaço livre pela HSSI é utilizado para alocação caixas ainda não empacotadas.

A Figura 43 apresenta a solução final em vários ângulos do 7º teste.

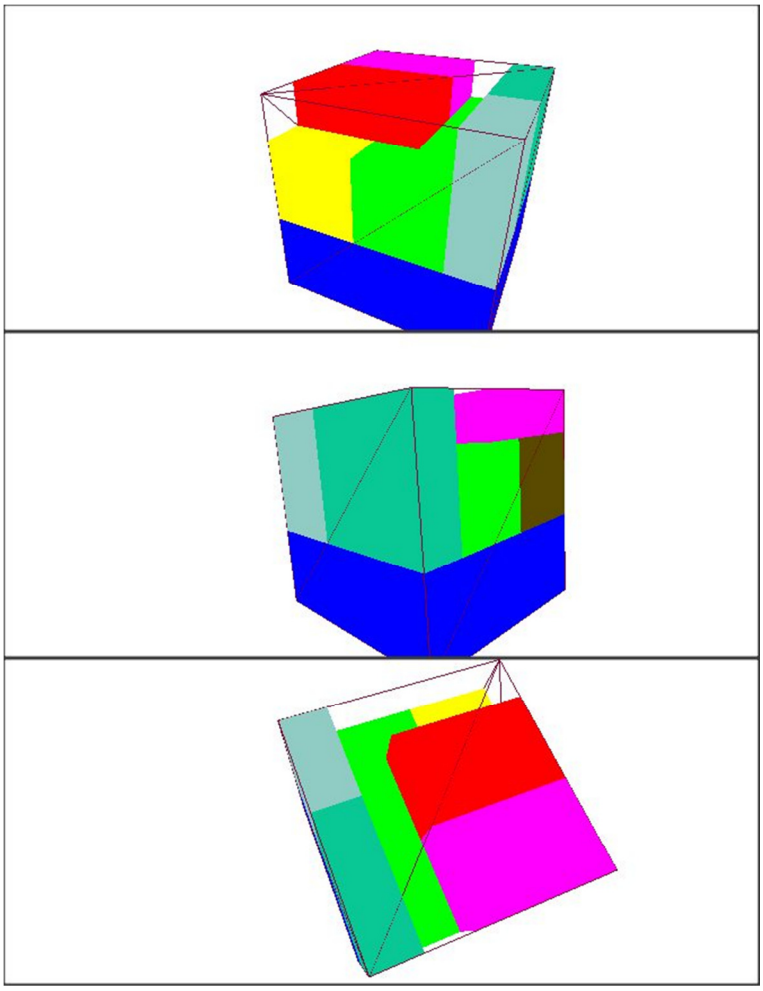


Figura 43 Teste 7-Resultado 2

Com este resultado conclui-se o desenvolvimento do algoritmo.

6. CONCLUSÕES

Ao longo deste texto foram sendo apresentadas conclusões que permitiram sustentar as opções de desenvolvimento do algoritmo genético. A primeira consideração final que se pode colocar refere-se, ao facto do algoritmo genético ser muito pouco eficiente quando utiliza um gerador de população inicial ou operadores de evolução aleatórios. Esta ineficiência pode ser contornada com a introdução de mecanismos que controlam a geração de soluções iniciais e os operadores genéticos, como por exemplo heurísticas.

No algoritmo desenvolvido foi utilizado a heurística HSSI para controlar a população inicial, garantindo assim que qualquer solução presente na população inicial é viável e não existe repetição entre as várias soluções iniciais.

Os resultados obtidos com esta heurística foram apreciáveis, verificando-se que estes eram viáveis e interessantes do ponto de vista da racionalização do espaço. Contudo a técnica que a HSSI utiliza para formar padrões de empacotamento, leva a que não exista uma combinação de espaços livres entre as várias camadas, ignorando assim possíveis alocações de caixas ao espaço livre formado pela agregação dos vários espaços livres de cada camada.

Esta limitação da HSSI foi ultrapassada através da introdução de um operador mutação que analisa o espaço livre toral do contentor e caso seja possível irá afetar uma caixa a esse espaço. Esta afetação é feita de forma a preencher o mais possível o espaço através da alocação de uma ou varias caixas.

Esta nova capacidade introduzida no algoritmo genético permitiu que algumas soluções fossem otimizadas, mais precisamente aquelas em que a HSSI preenche todo o espaço livre, ou quando existiam caixas ainda não empacotadas.

Como resultado final o algoritmo apresenta soluções interessantes mesmo para problemas onde existe objetos fortemente heterogéneos. Contudo a introdução de mecanismos que controlam a geração da população inicial limitam a diversificação das possíveis soluções podendo na prática eliminar soluções mais interessante que as apresentadas como resultado final. No entanto esta contra partida traz a vantagem de ser possível obter uma boa solução num curto espaço de tempo.

O *software* desenvolvido para introduzir a configuração do problema e apresentar a solução obtida pelo algoritmo revelou-se, uma ferramenta prática e de simples interpretação para utilizar em problemas de empacotamento onde existe apenas um contentor.

Uma possível melhoria que pode ser introduzida futuramente neste algoritmo é a capacidade para devolver soluções onde existe vários contentores, de dimensões iguais ou diferentes entre si. Esta nova funcionalidade iria permitir tornar esta ferramenta mais abrangente para problemas de empacotamento.

Em sumula o resultado final deste trabalho é positivo dado que foram atingidos os objetivos estipulados inicialmente, conseguido tornar mais clara a dificuldade que este problema apresenta do ponto de vista da obtenção de uma solução ótima.

Referências Documentais

[Leonid, 1939] LEONID.V. K. Mathematical Methods of Organizing and Planning Production. Management Science, 6, 1960, pp. 366–422.

[Gilmore e Gomory, 1961] GILMORE, P. e GOMORY, R. A linear programming approach to the cutting stock problem. Operations Research, 9, 1961, pp. 849-859.

[Gilmore e Gomory, 1963] GILMORE, P. e GOMORY, R. A linear programming approach to the cutting-stock problem II. Operations Research, 11, 1963, pp. 863-888.

[Gilmore e Gomory, 1965] GILMORE, P. e GOMORY, R. Multistage cutting stock problems of two and more dimensions". Operations Research, 14, 1965, pp. 1045-1074.

[Haessler, 1971] HAESSLER, R. W. A heuristic programming solution to a nonlinear cutting stock problem, Management Science, 17, 1971, pp. 793-802.

[Haessler, 1975] HAESSLER, R. W. Controlling cutting pattern changes in one-dimensional trim problems, Operations Research, 23, 1975, pp. 483-493.

[Garey e Johnson, 1979] GAREY, M. R. & JOHNSON, D. S. Computers and Intractability: a guide to the theory of NP-Completeness. San Francisco, Freeman, 1979.

[Bischoff e Marriott, 1990] BISCHOFF, E. E. e MARRIOTT, M. D. A comparative evaluation of heuristics for container loading. European Journal of Operational Research, 44, 1990, pp. 267–276.

[Goulimis, 1990] GOULIMIS, C. Optimal solutions to the cutting stock problem". European Journal of Operational Research, 44, 1990, pp. 197-208.

[Wäscher et al., 2007] WÄSCHER, G., HAUBNER, H., SCHUMANN, H. An improved typology of cutting and packing problems. European Journal of Operational Research 183, 2007, pp. 1109-1130.

[Aloise, Gonzaga e Bittencourt, 2003] ALOISE, D. J.; GONZAGA, C. S; BITTENCOURT, V. G. Uma Heurística de Suavização de Superfícies Irregulares para a solução do Problema Bin Packing 3D. Natal-RN, Brasil, 2003.

[Aloise, Gonzaga e Bittencourt, 2003] ALOISE, D. J.; GONZAGA, C. S; BITTENCOURT, V. G. Uma Heurística de Suavização de Superfícies Irregulares para a solução do Problema Bin Packing 3D. Natal-RN, Brasil, 2003.

[Mayr, 1984] MAYR, ERNST. Typological versus Populational Thinking" in Sober, E. (org). Conceptual Issues in Evolutionary Biology. London: MIT/Cambridge, 1984, pp. 14-17.

[Bortfeldt, 1994] BORTFELDT, A. A genetic algorithm for the container loading problem. Proceedings of the

Conference on Adaptive Computing and Information Processing, London, Vol. 2, 1994, pp. 25-32.

[Bortfeldt e Gehring, 1997] GEHRING, H., A. BORTFELDT. A Genetic Algorithm for Solving the Container Loading Problem. Internat. Trans. in Oper. Res. 4, 1997, pp. 401–418.

[Bortfeldt e Gehring, 1998] BORTFELDT, A.; GEHRING, H. Ein Tabu Search-Verfahren für Container belade probleme mit schwach heterogenem Kistenvorrat. OR Spektrum, 20, 1998, pp. 237-250.

[Bortfeldt e Gehring, 2001] GEHRING, H., A. BORTFELDT. A Parallel Genetic Algorithm for Solving the Container Loading Problem. Internat. Trans. in Oper. Res. 9, 2002, pp. 497-511.

[Bischoff e Marriott, 1990] BISCHOFF, E. E.; MARIOTT, M. D. A comparative evaluation of heuristics for container loading. European Journal of Operational Research, 44, 1990, pp. 267 - 276.

[Bischoff et al., 1995] BISCHOFF, E. E., JANETZ, F. & RATCLIFF, M. S. W. Loading pallets with non-identical items. European Journal of Operational Research, 84, 1995, pp. 681-692.

[Davies e Bischoff, 1999] DAVIES, A. P.; BISCHOFF, E. E. Weight distribution considerations in container loading. European Business Management School, University of Wales, Swansea, Statistics and OR Group, working paper

[Bischoff e Ratcliff,1995] BISCHOFF, E.E. & RATCLIFF, M.S.W. Issues in the development of approaches to container loading. Omega, 23, 1995, pp. 377-390.

[Bischoff, 2003] BISCHOFF, E. E. "Dealing with load bearing strength considerations in container loading problems". Technical Report. European Business Management School University of Wales. Swansea, 2003.

[Pisinger, 2002] PISINGER, D. Heuristics for the container loading problem. European Journal of Operational Research, 141, 2002, pp. 382-392.

[Scheithauer, 1991] G. SCHEITHAUER, A note on handling residual length. Optimization, 22, 1991, pp. 461-466

[Whitley et al., 1989] WHITLEY, D., STARKWEATHER, T., AND FUQUAY, D. Scheduling problems and traveling salesman: The genetic edge

recombination operator. In Proceedings of the Third International Conference on Genetic Algorithms. Los Altos, CA: Morgan Kaufmann Pulbishers, 1989.

[Eley, 2002] ELEY,M. Solving container loading problems by block arrangement. European Journal Of Operational Research, 141, 2002, pp. 393-409.

[Ngoi e Whybrew, 1993] NGOI, B.K.A; WHYBREW, K. A fast spatial representation method. Journal of International Advance Manufacturing Techology. 1993.

[Lipovetskji, 1988] LIPOVETSKJI, A. I. "Properties of the allocations of rectangles and some algorithms for optimal cutting layout". Preprint of Ural Branch of Acad. Sci. of USSR. Sverdlovsk, 1988.

[Bäck et al., 1991] BÄCK, T., HOFFMEISTER, F., AND SCHWEFEL, H. A survey of evolution strategies. In Proceedings of the Fourth International Conference on Genetic Algorithms, 1991, pp. 1-10.

[Schwefel, 1995] Schwefel, H-P. Evolution and Optimum Seeking, Wiley, New York, NY, 1995.

[Holland, 1975] Holland, J. Adaptation In Natural and Artificial Systems. The University of Michigan Press, Ann Arbour, 1975.

[Goldberg, 1989] Goldberg, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, 1989.

[Eiben e Smith, 2003] A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing, Springer, 2003.

[De Jong, 1975] DeJong, K. A. An Analysis of the Behavior of a class of Genetic Adaptive Systems. PhD thesis, University of Michigan, Ann Arbour. Department of Computer and Communication Sciences, 1975.

[Mitchell et al., 1997] Van Nimwegen, E., Crutchfield, J. P., and Mitchell, M. Finite populations induce metastability in evolutionary search. Physics Letters A, 229, 1997, pp. 144-150.

[Reeves, 1995] Reeves, C. R. Modern Heuristic Techniques for Combinatorial Problems, McGraw-Hill Book Company Europe, 1995.

[Gomes et al. 2004] Correia, M. H., A. M. Gomes, J. F. Oliveira and J. S. Ferreira. Problemas de empacotamento tridimensional. Investigação Operacional 12, 1992, pp. 169-180.

[De Jong e Sarma, 1992] DE JONG, KENNETH A. AND JAYSHREE SARMA, Foundations of Genetic Algorithms – 2, Darrell Whitley, pp. 19-28.

[Loh and Nee, 1992] LOH, T.H. AND A.Y.C. NEE. A packing algorithm for hexahedral boxes. Proceedings of the Conference of Industrial Automation, (CIA' 92),
Singapore, 1992, pp. 115-126.

[Rodrigues, 2000] Rodrigues, Rosiane de Freitas. Complexidade Computacional – NP-Completude. ICE – Departamento da Ciência da Computação, 2000.

[Ngoi et al., 1994] Ngoi, B.K.A., M.L. Tay and E.S. Chua. Applying spatial representation techniques to the container packing problem. Int. J. Prod. Res., 32, 1994, pp. 111-123.

[Faina, 2000] Faina, L. A global optimization algorithm for the three-dimensional packing problem. European Journal of Operational Research, 126, 2000, pp. 340-354.

[Baker, 1985] Baker, J.E. Adaptive selection methods for Genetic algorithms. In Proceedings of the 1st International Conference on Genetic Algorithms and their Applications, 1985, pp. 101-111

[Baker, 1987] Baker, J. E. reducing Bias and Inefficiency in the selection Algorithm. Proceedings of the Second International Conference on Genetic Algorithms, Lawrence Erlbaum, Hillsdale, NJ, 1987, pp. 14-21.