ESCOLA POLITÉCNICA
DE PERNAMBUCO

d.s.c. UPE
Departamento de Sistemas e Computação

SIP – Sistema Inteligente de Carregamento de Paletes

Trabalho de Conclusão de Curso

Engenharia da Computação

Nome do Aluno: George Moraes Cavalcanti Júnior
Orientador: Prof. Carmelo José Albanez Bastos Filho

UNIVERSIDADE
DE PERNAMBUCO

George Moraes Cavalcanti Júnior

SIP – Sistema Inteligente de Carregamento de Paletes

Monografia apresentada como requisito parcial para obtenção do diploma de Bacharel em Engenharia da Computação pela Escola Politécnica de Pernambuco – Universidade de Pernambuco.

Recife, junho de 2009.

Dedico este trabalho ao meu avô Joaquim, pelo grande coração deste homem, ele contribui para minha vida das mais diversas formas.

Agradecimentos

Agradeço a Deus, em primeiro lugar, por sua maravilhosa graça, misericórdia e força que me conduz e me apóia na caminhada da vida pessoal e acadêmica. Ele é meu maior sustento.

À minha família, principalmente, aos meus pais, George e Fabíola, os quais têm me dado todo o apoio e encorajamento nas minhas decisões, pelos seus cuidados e pela luta deles para me proporcionar a melhor educação. Também à minha irmã e companheira Lays e aos meus avós, Joaquim e Dinalva, pelos diversos incentivos e contribuições deram para a minha vida.

Aos meus amigos da Igreja Batista Koinonia por todos os conselhos, orações e ajudas de diversas formas. Agradeço por todo o apoio que recebi deles.

Agradeço aos meus amigos de curso, pessoas fundamentais para compartilhar momentos agradáveis ou dividir as dificuldades ao longo da academia. Suas contribuições são altamente importantes para minha formação. Dentre eles, agradeço principalmente a Carlos Bruno, Gustavo Henrique e Maira Sabóia.

Aos professores, especialmente a Carmelo Bastos, que me orientou de forma impecável e foi sempre presente durante as atividades de orientação. Agradeço por toda prontidão e disposição em ajudar na execução deste trabalho. Sua colaboração acrescentou significativamente na minha formação acadêmica.

Aos companheiros de trabalho. Ao supervisor Daniel Brasil, cuja competência e prontidão em auxiliar contribuíram bastante para minha formação profissional e acadêmica. Também o agradeço pelas sugestões fornecidas para este trabalho.

Resumo

Em logística há uma serie de operações que são executadas para levar um produto da fábrica até o vendedor. Tais operações têm diversos custos associados, os quais terão impacto no preço de venda dos produtos. Uma dessas operações é a paletização, a qual consiste em colocar produtos sobre uma plataforma (denominada palete) para facilitar a movimentação dessa carga. Em muitos casos esses produtos não são colocados de forma que não utilizam o espaço possível disponível no palete. Isso gera maior custo em operações de transporte e armazenagem. Esse problema é conhecido como problema de carregamento de paletes. Ele consiste em organizar as caixas de produtos sobre o palete visando otimizar a ocupação de área disponível. Para resolver este tipo problema, foi desenvolvido neste trabalho uma ferramenta computacional que mostra posicionamento das caixas no palete, otimizando seu uso. Esta ferramenta utiliza Algoritmos Genéticos para atingir seu propósito, que é auxiliar na resolução de problemas de carregamento de paletes. Para certos casos do problema, a ferramenta alcançou a solução ótima. No entanto, para casos mais complexos a solução ótima não foi encontrada nos experimentos realizados.

Abstract

In logistics there are many operations that are performed to transport a product from the factory to the seller. Such operations have different costs that will impact on the sale price of the products. One operation is the palletization, which is to place products on a platform (called pallet) to facilitate the movement of the cargo. In many cases, such organization process is not optimized. Therefore, the available space not used on the pallet generates higher costs for transportation and storage operations. This problem is known as pallet loading problem. It consists to organize boxes on the pallet to optimize the occupation of the available area. To solve this problem, a computational tool was developed in this work that to show the best position of boxes on pallet, optimizing its use. This tool uses Genetic Algorithms to achieve its purpose, which is to solve pallet loading problems. In some cases, the tool reached the optimal solution. However, in some complex cases, the optimal solution was not found.

Sumário

Índice de Figuras

Índice de Tabelas

Tabela de Símbolos e Siglas

PBR – Padrão Brasileiro

PLP – Pallet Loading Problem (Problema de carregamento de paletes)

SIP – Sistema Inteligente de Carregamento de Palete

TSP – Traveling Salesman Problem (Problema do Caixeiro Viajante)

API – Application Programming Interface (Interface de Programação de Aplicativos)

DSC – Departamento de Sistemas e Computação

Capítulo 1

Introdução

A globalização e a evolução no processo de produção trouxeram para as empresas a necessidade de maior agilidade no manuseio de cargas e de diminuição nos custos nas operações de transporte e armazenagem de mercadorias. Essas empresas precisam distribuir uma enorme variedade de produtos para lugares distantes. No entanto, o processo de distribuição não é um fator simples. Além disso, os custos envolvidos se refletem nos preços finais de venda dos produtos. Nesse processo, as mercadorias são manuseadas várias vezes e, geralmente, transportadas em mais de um modal até chegar ao destino. Modal é a modalidade de transporte que se utiliza para movimentar mercadorias/produtos, ou seja, transporte rodoviário, ferroviário, aéreo e aquaviário (fluvial e marítimo). Sendo assim, um mesmo conjunto de mercadorias pode passar diversas vezes por processos de carga e descarga até chegar a seu destino. Com o aumento no manuseio das mercadorias, fez-se necessário padronizar os equipamentos de maneira que as mercadorias produzidas pudessem ser facilmente movimentadas ao longo de sua cadeia logística. Com isso, surgiram os métodos de unitização, que consistem em agrupar vários volumes heterogêneos em uma unidade de carga, em geral com dimensões padronizadas. Unir pequenos volumes heterogêneos em grandes volumes homogêneos traz ganhos de produtividade significativos em questões de tempo, espaço e custo mediante a utilização de unidades de carga, como, por exemplo, os paletes.

O palete é uma plataforma de madeira, metal ou plástico utilizado para a movimentação de cargas. Sua função é facilitar o transporte de cargas através da utilização de empilhadeiras e paleteiras. A popularização dessas unidades trouxe uma nova problemática para as operações logísticas: a necessidade de otimizar a ocupação do espaço disponível nos paletes devido aos custos de armazenagem e transporte (frete) de mercadorias. Um pequeno aumento no número de caixas de mercadorias estocadas em um palete pode resultar numa economia significativa em termos globais. Isso devido à grande quantidade de produtos que são estocados e

transportados mundialmente. A Figura 1 mostra um palete denominado PBR (Padrão Brasileiro), é feito de madeira, com medidas 1,20m x 1,00m e é bastante popular nas operações logísticas. Há outros padrões de paletes com outras dimensões. A padronização é para as dimensões do palete e independe do material que o constitui.

Figura 1. Palete PBR de madeira.

Várias pesquisas têm sido desenvolvidas com objetivo de otimizar a utilização dos paletes. Esse problema é conhecido como o problema de carregamento de paletes [1] (*Pallet Loading Problem – PLP*). Ao estudá-lo, Hodgson [2], o dividiu em dois casos: o problema de carregamento de paletes do produtor (*manufacturer's pallet loading problem*) e o problema de carregamento de paletes do distribuidor (*distributor's pallet loading problem*).

Este trabalho focará no primeiro caso. Neste, o produtor fabrica produtos a serem empacotados em caixas iguais de dimensões (*c, l, h*) (*c* é o comprimento da caixa, *l* é a largura da caixa e *h* é a altura da caixa) as quais serão colocadas em paletes de dimensões (*C, L, H*) (*C* é o comprimento do palete, *L* é a largura do palete e *H* é a altura máxima aceitável para o carregamento do palete). O problema consiste em arranjar ortogonalmente o máximo número de retângulos (*c*, *l*) (i.e. base das caixas) dentro de outro retângulo (*C*, *L*) (i.e. superfície do palete).

Na literatura encontram-se diversas abordagens em busca de melhores abordagens do *PLP*, desde algoritmos exatos [3] até criação de heurísticas [4] e aplicações de metaheurísticas (*Tabu Search* [5][6], *Simulated Annealing* [7] *e* Algoritmos Genéticos [8]). Heurística é uma regra, simplificação, ou aproximação que reduz ou limita a busca por soluções em domínios difíceis e pouco conhecidos

de um determinado problema. Enquanto que uma metaheurística é um método heurístico para resolver, de forma genérica, problemas de busca e otimização.

1.1 Motivação

O processo de paletização, pelo qual volumes (caixas, sacos, tambores, etc.) são colocados sobre um palete, é altamente empregado nas operações logísticas e grande facilitador na movimentação de cargas. Ele deve ser executado com eficácia para atingir os resultados esperados, que seria colocar a maior quantidade de volumes sobre o palete (conseqüentemente ocupando maior área na superfície) respeitando o limite de altura (*H*) para o carregamento, além de atender critérios de estabilidade do conjunto [9].

Para alcançar tal objetivo, pode-se definir um *layout* do posicionamento das caixas baseando-se na experiência de um carregador ou utilizar um *software* que retorne esse *layout*, usando, por exemplo, as medidas do palete e das caixas como dados de entrada. A primeira solução pode ser mais simples e viável para casos em que o carregamento comporta poucas caixas, ou seja, com caixas grandes. No entanto, se as caixas forem pequenas (geralmente mercadorias com alto valor agregado como fármacos e dispositivos eletrônicos) aumenta a dificuldade para definição do melhor posicionamento, devido à maior quantidade de caixas que serão acondicionadas sobre o palete. A Figura 2 ilustra um exemplo de carregamento com 133 caixas idênticas (as áreas hachuradas representam as áreas não ocupadas pelas caixas). Para estes tipos de carregamentos dificilmente um carregador irá conseguir dispor em tempo hábil as caixas de maneira que otimize a utilização da área do palete. A dificuldade ainda é maior se o carregamento puder conter caixas de tamanhos diferentes, mas tal caso não pertence ao escopo deste trabalho.

Normalmente, as atividades de carregamento são realizadas sem considerar a otimização do uso do palete. Os carregadores fazem a paletização, geralmente, por tentativa e erro, o que não garante que foi definido um carregamento com a maior quantidade de caixas.

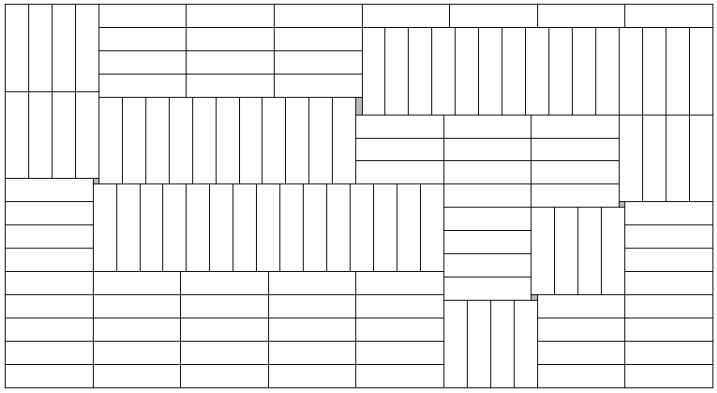

Figura 2. *Layout* de caixas para uma instância do problema.
Palete (121 x 66) com 133 caixas (15 x 4).

O dimensionamento das embalagens secundárias é outro quesito bastante relevante quando se trata da paletização. Embalagens secundárias são as utilizadas para facilitar a distribuição e comercialização, agrupando certa quantidade de embalagens primárias ou unitária, as quais contêm o produto. As dimensões das embalagens secundárias irão impactar diretamente na otimização do uso do palete, pois, geralmente, elas que são carregadas nos paletes. Sendo assim, o projeto de uma embalagem deve considerar as medidas que irão gerar melhor aproveitamento do palete.

Das questões relativas à paletização de caixas é que surge a necessidade de uma ferramenta que possa ajudar no processo de paletização. O programa poderá levar um ganho de produtividade para as operações logísticas, bem como contribuir no projeto de embalagens. Em primeira instância gerando o *layout* das caixas, o que deve agilizar o processo de carregamento, ainda otimizando o uso do palete, e em segunda instância aumentando o percentual de área utilizada no palete, o que pode levar a escolha de medidas que cubram maior área e conseqüentemente contenha mais produtos. Em ambos os casos, a ferramenta levará à redução de custos logísticos relativos a transporte e armazenagem.

1.2 Objetivos

Com a execução deste trabalho pretende-se investigar e desenvolver algoritmos para PLP e em seguida criar uma ferramenta. Esta ferramenta deve gerar os *layouts*

de organização das caixas sobre o palete, mostrar o percentual de área ocupada do palete e retornar a quantidade de caixas que foram paletizadas.

O trabalho está dividido em duas fazes. A primeira fase é dirigida à investigação de algoritmos e das maneiras de se tratar o problema. Sendo assim, busca-se obter o bom entendimento dessas abordagens, visando compreender suas vantagens e desvantagens. Esses aspectos estão relacionados, principalmente, à qualidade das soluções encontradas, ao tempo de convergência e à capacidade de resolver as diferentes classes de instâncias do problema.

Após a compreensão do contexto do PLP e suas abordagens, segue-se para a segunda fase, o desenvolvimento da ferramenta. As principais entradas desse sistema são as medidas das caixas e do palete. De posse dessas medidas, o algoritmo implementado gera uma solução para o problema. Uma solução é a maior quantidade de caixas que o algoritmo conseguiu por sobre o palete, isto é, a quantidade de retângulos (c, l) que foram postas no interior de outro retângulo (C, L).

Para atingir o propósito de otimizar o carregamento (i.e. aproveitar a maior área possível do palete), foi utilizada uma técnica de computação evolucionária, especificamente Algoritmos Genéticos. A função objetivo é minimizar a quantidade de sobreposições das caixas, pois nas soluções iniciais as caixas são dispostas aleatoriamente e é possível que elas se sobreponham. Quando a quantidade de sobreposições é 0, então o algoritmo retorna tal solução. Através dos operadores empregados nessa técnica será possível diversificar as soluções e assim explorando o espaço de busca por melhores soluções.

Poucas abordagens do PLP utilizam alguma técnica de inteligência computacional. Então, também se espera com este projeto desenvolver a ferramenta de maneira que ela possa, com devidas adaptações, ser incorporada ao ambiente de produção de transportadoras e operadores logísticos. Ela poderá render ganhos de produtividade nas atividades de paletização de cargas e contribuir para a redução de custos logísticos relacionados a transporte e armazenagem.

1.3 Estrutura do Trabalho

Este trabalho aborda os aspectos relacionados ao PLP, Algoritmos Genéticos e o desenvolvimento da ferramenta SIP (Sistema Inteligente de Carregamento de Palete). Sua ênfase está em descrever como os Algoritmos Genéticos podem ser empregados em problemas de carregamento de paletes.

O Capítulo 2 fornece uma visão geral de algumas abordagens utilizadas para resolver o PLP, bem como explica os conceitos relacionados.

O Capítulo 3 explica detalhadamente os Algoritmos Genéticos, a representação dos indivíduos, o uso dos operadores e o fluxo de execução do algoritmo. Ao longo desse capítulo, a maior parte dos conceitos é exemplificada.

O Capítulo 4 mostrará os recursos da ferramenta desenvolvida e explicará como o problema foi modelado para incorporar o uso dos Algoritmos Genéticos.

O Capítulo 5 apresentará os experimentos realizados e os resultados obtidos com a ferramenta desenvolvida, assim como trará uma análise dos mesmos.

O Capítulo 6 trará a conclusão do trabalho, onde será exposta uma avaliação sobre os resultados obtidos ao final do trabalho. Serão mostradas algumas contribuições decorrentes deste trabalho e também expostas algumas dificuldades encontradas ao longo do seu desenvolvimento. Por fim, serão citados alguns trabalhos futuros que poderão ser executados, dando continuidade à ferramenta SIP.

Capítulo 2

Problema de Carregamento de Paletes

Neste capítulo será exposta uma visão geral sobre problema de carregamento de palete e uma breve explicação sobre heurística de blocos.

A grande preocupação com a redução de custos das atividades logísticas tem conduzido pesquisas para o problema de carregamento de paletes. Esse problema consiste em arranjar, ortogonalmente e sem sobreposição, o máximo número de caixas retangulares com dimensões (c, l, h) sobre um palete de dimensões (C, L, H), onde H é a altura máxima tolerável no carregamento. Tal problema é tido como uma classe dos problemas de Corte e Empacotamento [10]. O problema de carregamento de paletes aparece nas operações logísticas de transporte, distribuição e armazenamento de produtos embalados em caixas. É importante otimizar o uso de cada palete, pois mesmo com aumento de poucas caixas no carregamento, pode resultar em economias significativas na execução dessas operações.

Neste trabalho é abordado o problema de carregamento de paletes do produtor. Nele é tratado o carregamento de produtos embalados em caixas idênticas. Em cada camada horizontal de carregamento, uma orientação vertical das caixas é fixada. Já no problema de carregamento de paletes do distribuidor é possível criar carregamentos com caixas diferentes, o que o torna mais complexo do que o problema do produtor. Este tipo problema é similar ao problema de carregamento de contêineres, nos quais o carregamento é bastante complexo como mostra a Figura 3.

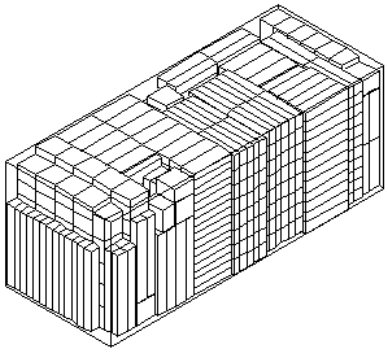

Figura 3. Carregamento de contêiner.

2.1 Heurística de blocos

Nas heurísticas de bloco [11] o palete é dividido em retângulos nos quais só poderão ser colocadas caixas na mesma orientação (horizontal ou vertical), ou seja, são construídos padrões de carregamento compostos por um ou mais blocos cujas caixas possuem a mesma orientação.

Existem determinadas classificações para padrões de carregamento, são eles:

- Guilhotinado: Ao ser executado sobre um retângulo, produz dois novos retângulos. Um padrão obtido por sucessivos cortes guilhotinados é chamado padrão guilhotinado. Esse padrão é mostrado na Figura 4.
- Padrão não guilhotinado de primeira ordem: Ao ser produzido sobre um retângulo, produz cinco novos retângulos arranjados de maneira a não formar um padrão guilhotinado, como mostrado na Figura 5. Um padrão obtido por sucessivos cortes guilhotinados e/ou cortes não guilhotinados de primeira ordem é chamado padrão não guilhotinado de primeira ordem. Todo padrão guilhotinado também é um padrão não guilhotinado de primeira ordem, porém o contrário não é verdade.
- Padrão não-guilhotinado de ordem superior: É um padrão que não pode ser obtido através de sucessivos cortes guilhotinados e/ou cortes não guilhotinados de primeira ordem. Esse padrão é mostrado na Figura 6.

As heurísticas de bloco podem gerar soluções ótimas para os casos em o que carregamento ótimo é um padrão não guilhotinado de primeira ordem, pois só geram soluções desse tipo. Para problemas cuja solução ótima é um padrão de ordem superior, as heurísticas de bloco não encontram a solução ótima.

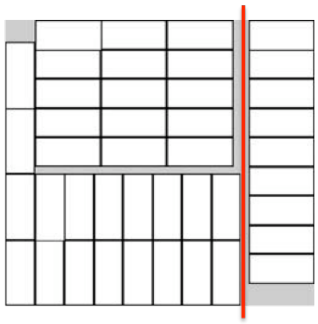

Figura 4. Padrão guilhotinado.

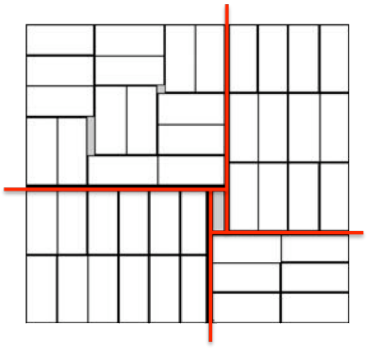

Figura 5. Padrão não guilhotinado e corte em negrito.

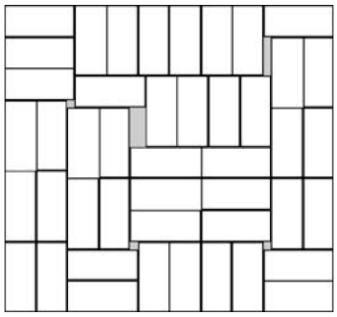

Figura 6. Padrão não guilhotinado de ordem superior.

Existem várias abordagens que fazem uso de metaheurísticas para a resolução de problema de carregamento de paletes, dentre elas estão: *Tabu search*, *Simulated Annealing* e Algoritmos Genéticos. Elas não limitam as soluções a determinado tipo de carregamento, sendo assim este trabalho visa aplicar os Algoritmos Genéticos para resolver esse problema.

Capítulo 3

Algoritmos Genéticos

Historicamente, as primeiras iniciativas na área da Computação Evolutiva foram de biólogos e geneticistas interessados em simular os processos vitais de um ser humano em um computador, na época (final da década de 50) foram denominados de “processos genéticos”. Na década de 60, o cientista *John Holland* iniciou um estudo [12], no qual era implementado uma população de indivíduos os quais possuíam cromossomos e eram sujeitos a operações de seleção, mutação e cruzamento. Desse estudo foram criados os Algoritmos Genéticos.

Os Algoritmos Genéticos são métodos generalizados de busca e otimização que simulam os processos naturais de evolução, baseados nas idéias de seleção natural propostas por *Charles Darwin* [13], visando aplicá-las na resolução de problemas reais, como maximização e minimização de funções. Nessas simulações há uma população de indivíduos, os quais possuem uma constituição genética (cromossomo) que são codificações de soluções de um problema. Os indivíduos serão selecionados com o passar das gerações (iterações do algoritmo) em busca de melhores soluções. Geralmente, o conjunto de soluções iniciais é criado aleatoriamente e a cada geração esse conjunto é evoluído. Para simular o efeito de evolução, utilizam-se critérios de avaliação e seleção dos indivíduos. O primeiro determina a aptidão (ou *fitness*), enquanto que o segundo seleciona, de forma determinística ou probabilística, os indivíduos com maior aptidão que irão compor as gerações futuras. Assim, a tendência é que ao longo das gerações os indivíduos mais aptos sobrevivam e conseqüentemente ocorra a convergência para uma solução ótima (ou aproximadamente ótima) do problema, já que cada indivíduo representa uma solução.

Para cada tipo de problema pode haver uma maneira específica de se avaliar os indivíduos. Tomando um simples problema como exemplo: minimizar a função quadrática $f(x) = x^2$ no domínio $D(f) = [0,15]$. Para esse caso, um número real

pertencente ao domínio da função pode ser um possível cromossomo de um indivíduo. Certas abordagens de Algoritmos Genéticos codificam o cromossomo dos indivíduos de forma binária. Supondo uma população inicial de 2 indivíduos, na qual cada um possui um cromossomo que é um número binário. A avaliação dos indivíduos é realizada de acordo com o Algoritmo 1.

Algoritmo 1. Avaliação dos indivíduos.

1. Converter valor do cromossomo para número decimal;
2. Usar o número encontrado como argumento (x) da função
3. Usar o valor de $f(x)$ como aptidão do indivíduo

A Tabela 1 mostra um exemplo com cromossomos dos 2 indivíduos da população, gerados aleatoriamente, e os valores correspondentes de x e $f(x)$. Como este exemplo trata de um problema de minimização, o indivíduo de G1 é considerado mais apto que o G2, pois seu cromossomo corresponde a um valor mais próximo ao mínimo da função.

Tabela 1. Cromossomos de indivíduos e função de avaliação.

	Cromossomo	x	$f(x)$
G1	0001	1	1
G2	0101	5	25

A função que se deseja otimizar no problema é denominada função objetivo (ou *fitness function*). No exemplo descrito esta função era simples e conhecida ($f(x) = x^2$), no entanto os Algoritmos Genéticos são, geralmente, aplicados em problemas complexos de otimização. A grande vantagem dos Algoritmos Genéticos está no fato de não ser necessário saber como funciona a função objetivo, é preciso apenas tê-la disponível para aplicar aos indivíduos, comparar os resultados e conhecer o formato das entradas. Sendo assim, as aplicações são para problemas com diversos parâmetros ou características que precisam ser combinadas em busca da melhor solução; problemas com muitas restrições ou condições que não podem ser representadas matematicamente; e problemas com grandes espaços de busca.

Alguns exemplos são: Otimização Combinatória [14], Problema do Caixeiro Viajante (TSP - *Traveling Salesman Problem*) [15], Otimização de Síntese de Circuitos Eletrônicos [16], dentre outra aplicações.

Essa gama de aplicações para os Algoritmos Genéticos não seria possível apenas com os critérios que definem a aptidão dos indivíduos e o operador de seleção. Se utilizado somente esse operador a busca por soluções ficaria limitada apenas ao conjunto inicial de soluções, uma vez que não há alteração das soluções geradas. Por isso, outros dois operadores são empregados nos Algoritmos Genéticos, o operador de mutação e o de cruzamento (ou recombinação). O primeiro realiza alterações no cromossomo dos indivíduos e o segundo faz combinações entre dois indivíduos (os pais) gerando um terceiro (um filho) com o cromossomo constituído pela combinação dos cromossomos dos pais. Com isso, os novos cromossomos gerados, sejam por mutação ou por cruzamento, irão constituir novas soluções e, por conseguinte, irão diversificar o conjunto de soluções aumentando a capacidade de exploração do algoritmo e contribuindo para a busca por melhores soluções. O funcionamento desses três operadores (seleção, cruzamento e mutação) será detalhado na seção 3.2.

3.1 Representação dos Indivíduos

Como citado no início deste capítulo, um indivíduo é uma representação de uma solução. Eles possuem dois atributos básicos:

Cromossomo: Codificação de uma solução. O cromossomo deve ser uma representação capaz de cobrir todo o conjunto dos valores no espaço de busca, e deve ter tamanho finito. Geralmente o cromossomo é uma seqüência de bits. Além da representação binária, também se utilizam seqüências de números inteiros, números reais, letras, ou outros tipos de dados específicos de um problema abordado. Essas seqüências são, comumente, colocadas em *arrays*, pois facilitam as operações de cruzamento e mutação. Para problemas com múltiplas entradas é possível combinar as entradas (genes do indivíduo) em uma única seqüência de bits como mostra a Figura 7.

Cromossomo	0010101110011010 x y

Figura 7. Cromossomo binário com duas entradas.

- Aptidão (ou *fitness*): Qualidade da solução representada pelo indivíduo. É um atributo importante no momento de aplicação dos operadores sobre a população. Na seleção, os indivíduos mais aptos têm maior tendência de serem escolhidos do que os indivíduos menos aptos. No cruzamento, os indivíduos mais aptos terão maior possibilidade de se combinarem para gerar novas soluções. Por fim, na mutação, certa quantidade de indivíduos mais aptos não sofre a ação desse operador, para que as melhores soluções perdurem ao longo das gerações. Essa técnica é chamada de elitismo e será detalhada na seção 3.2.

3.2 Operadores

Nos Algoritmos Genéticos os operadores são fundamentais para o sucesso desse método de busca. Sem eles não seria possível realizar busca em amplitude e em profundidade no espaço de soluções:

- Busca em amplitude: consiste em visitar regiões que ainda não foram visitadas no espaço de busca, para tal é preciso criar diversidade no conjunto de soluções. Isso irá aumentar a capacidade de exploração (*exploration*) do algoritmo.
- Busca em profundidade: consiste em concentrar a busca em regiões mais promissoras do espaço de busca, ou seja, regiões onde se encontram boas soluções para o problema. Esse tipo de busca tende a refinar as soluções encontradas de maneira que elas convirjam para um ponto, podendo ser um máximo (ou mínimo) global ou local. A busca em profundidade está ligada à capacidade de prospecção (*exploitation*) do algoritmo.

Uma boa técnica de otimização deve combinar esses aspectos, *exploration* e *exploitation*, para encontrar a solução ótima (ou aproximadamente ótima) de um

problema (i.e. máximo ou mínimo global). Os Algoritmos Genéticos combinam ambos os aspectos através dos operadores de seleção (*exploitation*), mutação e cruzamento (*exploration*).

3.2.1 Cruzamento

O operador de cruzamento é conhecido também como operador de recombinação. Ele combina os cromossomos de dois indivíduos pais gerando um filho na população. Para descrever o funcionamento desse operador serão considerados dois indivíduos cujos cromossomos de tipo binário são mostrados na Figura 8. A diferença de cores nos cromossomos do PAI A e do PAI B é apenas para facilitar a visualização nos exemplos.

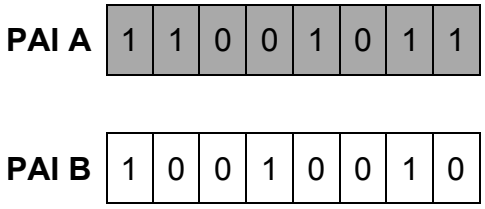

Figura 8. Indivíduos que serão cruzados.

Existem várias formas de realizar a operação de cruzamento. Três dessas formas serão explicadas:

- Cruzamento de ponto único: é escolhido um ponto de cruzamento aleatoriamente, em seguida é copiado parte do cromossomo de cada pai, depois as partes são concatenadas gerando o cromossomo de um filho. Uma parte desse filho é formada pelos bits que vão do primeiro até o ponto de cruzamento de um pai e pelos bits que vão do ponto de cruzamento até o último bit do outro pai. A Figura 9 ilustra esta operação.

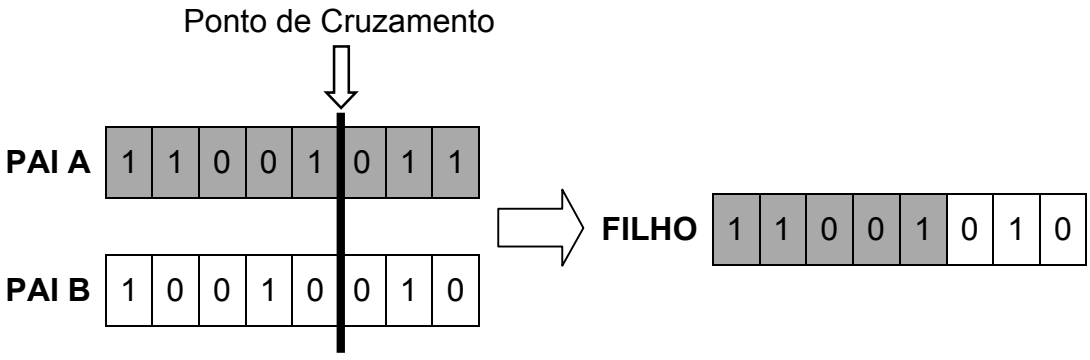

Figura 9. Cruzamento de ponto único.

- Cruzamento de dois pontos: são escolhidos dois pontos de cruzamento aleatoriamente, em seguida é copiado parte do cromossomo de cada pai, depois as partes são concatenadas gerando o cromossomo de um filho. Uma parte desse filho é formada pelos bits que estão dentro da faixa gerada pelos pontos de cruzamento em um pai e pelos bits que estão fora da faixa no outro pai. A Figura 10 ilustra esta operação.

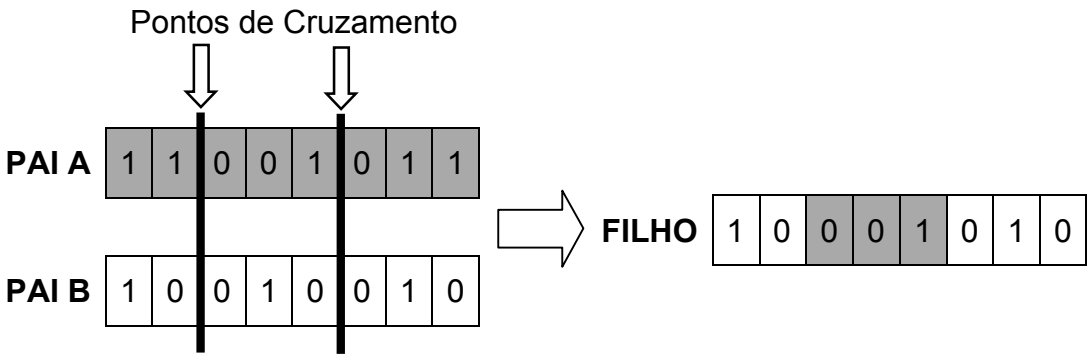

Figura 10. Cruzamento de dois pontos.

- Cruzamento uniforme ou multiponto: é gerado um indivíduo a partir de bits copiados aleatoriamente do primeiro ou do segundo pai. A Figura 11 ilustra esta operação.

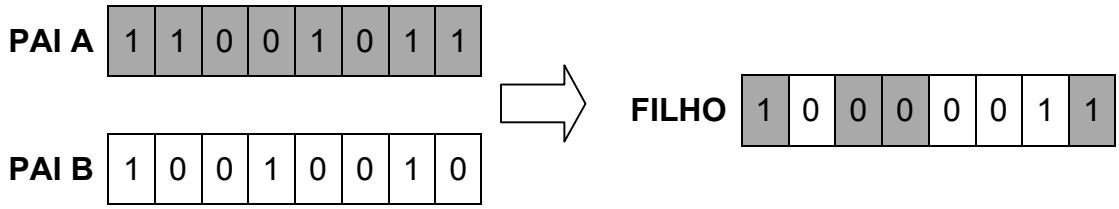

Figura 11. Cruzamento uniforme.

Em cada tipo de cruzamento há a opção de gerar um ou dois indivíduos (é uma escolha de implementação, que pode variar dependendo do problema). No caso de serem gerados dois indivíduos o segundo indivíduo será formado pelas partes que foram desprezadas de cada pai ao gerar o primeiro indivíduo.

3.2.2 Mutação

O papel principal do operador de mutação é mudar aleatoriamente parte das informações codificadas de um cromossomo para criar novas soluções. Por outro lado, a operação de mutação pode ser empregada para reordenar codificações

inválidas (não pertencentes ao espaço de busca), obtendo novas soluções viáveis e válidas.

No caso de indivíduos de cromossomo de tipo binário, a operação de mutação irá escolher alguns bits aleatoriamente e invertê-los. No caso de dos cromossomos de tipo inteiro ou real, quando um número é escolhido pra sofrer mutação este será substituído por outro número gerado aleatoriamente. A Figura 12 mostra a operação de mutação; os bits que têm preenchimento branco são os alterados.

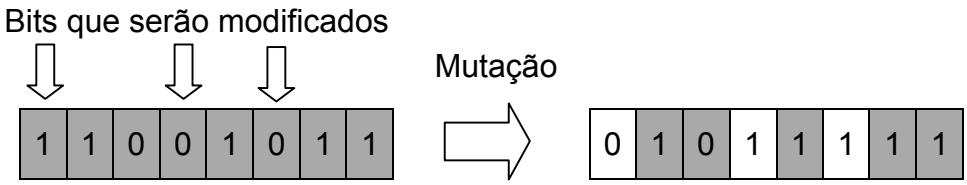

Figura 12. Operação de mutação.

3.2.3 Seleção

O operador de seleção é inspirado na idéia da seleção natural. Assim, a cada geração, os melhores indivíduos (mais aptos) são selecionados para gerar filhos através de cruzamento e mutação. Esse operador conduz as soluções para as melhores regiões do espaço de busca.

Um método seleção é o de Exclusão Social [17]. Primeiramente, a população é ordenada em ordem decrescente de aptidão, cada indivíduo irá cruzar com o indivíduo de aptidão imediatamente inferior a sua e gerar um filho. Em outras palavras, o indivíduo mais apto da população cruzaria com o segundo mais apto, o terceiro com o quarto e assim por diante. Poderia também ser da seguinte forma: o primeiro mais apto cruza com o segundo mais apto, este cruza com o terceiro mais apto e assim por diante. Neste trabalho, por decisão de projeto, os experimentos utilizaram apenas primeira abordagem descrita para a Exclusão Social.

Há outro método de seleção, este bastante típico, que é a seleção proporcional à aptidão, também conhecida como Seleção por Roleta. Os indivíduos são selecionados de forma probabilística. Cada um tem a probabilidade de ser

escolhido proporcionalmente à sua aptidão (i.e. quanto mais apto mais terá chance de ser selecionado). A Figura 13 mostra um exemplo de roleta com quatro indivíduos.

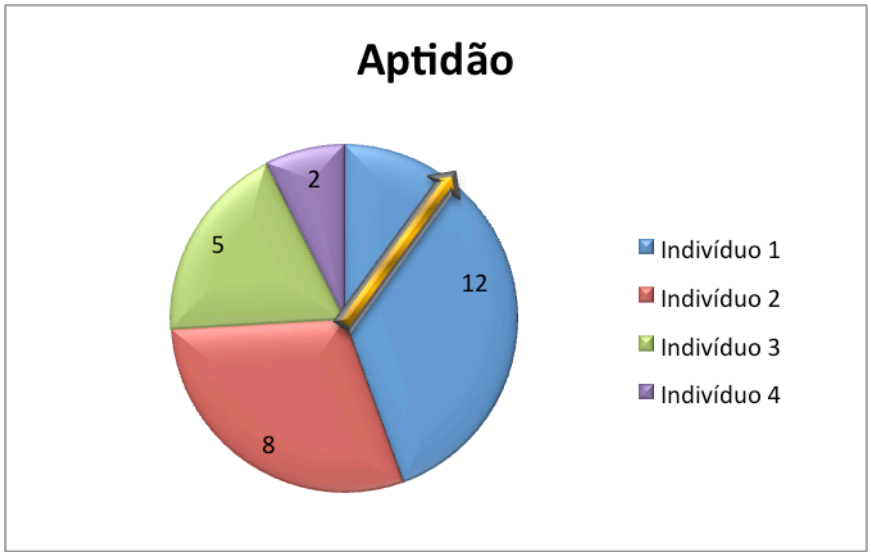

Figura 13. Roleta com seções proporcionais à aptidão dos indivíduos.

Ao "girar" esta roleta o indivíduo 1 terá a maior probabilidade de ser selecionado. Esse processo pode ser descrito segundo o Algoritmo 2.

Algoritmo 2. Pseudocódigo da roleta.

1. *Calcule a soma **T** dos valores de aptidão de todos os indivíduos da população.*
2. *Ordene a população de ordem decrescente de aptidão.*
3. *Gere um número aleatório **r** no intervalo* **[0,*T*]**.
4. *Percorra toda a população somando as aptidões dos indivíduos (soma **s**). Quando a soma **s** for maior que **r**, pare e retorne o indivíduo atual.*

3.2.4 Elitismo

O cruzamento e a mutação podem excluir o melhor indivíduo da população. Para evitar que a melhor (ou as melhores) solução seja descartada da população emprega-se essa técnica chamada elitismo [18]. Esse operador garante que os n melhores indivíduos sejam repassados para a geração seguinte.

3.3 Parâmetros dos Algoritmos Genéticos

Nesta seção serão explicados alguns parâmetros existentes nessa técnica:

- Tamanho da População: Determina o número de indivíduos da população inicial. É possível que a quantidade de indivíduos varie numa mesma geração devido às operações de cruzamento, mas ao final de cada geração a população deve estar com o mesmo tamanho da população inicial, ou seja, alguns indivíduos são descartados. O parâmetro em questão afeta diretamente o desempenho global e a eficiência dos Algoritmos Genéticos. Com uma população pequena, o desempenho pode ser insatisfatório, pois assim a população fornece pouca cobertura do espaço de busca do problema. Uma grande população geralmente fornece maior cobertura do domínio do problema, além de prevenir convergências prematuras para máximos (ou mínimos) locais ao invés de globais. No entanto, o aumento da população, gera um aumento no custo computacional do algoritmo.
- Taxa de Cruzamento: Quanto maior for esta taxa, mais rapidamente novos indivíduos serão introduzidos na população. Mas se esta for muito alta, a maior parte da população pode ser substituída, aumentando a possibilidade de ocorrer perda de indivíduos de alta aptidão. Com um valor baixo, o algoritmo pode tornar-se muito lento, afetando a velocidade de convergência, já que serão gerados poucos indivíduos novos a cada geração. Logo, reduz a capacidade de busca em amplitude.
- Taxa de Mutação: Determina a probabilidade com que uma mutação ocorrerá. Uma baixa taxa de mutação previne que um dado indivíduo fique estagnado no espaço de busca. Com uma taxa muito alta a busca se torna essencialmente aleatória além de aumentar muito a possibilidade de que uma boa solução seja destruída. A Taxa de Mutação é dependente da aplicação, mas para a maioria dos casos é entre 1 e 10%.
- Taxa de Elitismo: Determina a taxa de indivíduos mais aptos que serão repassados para a próxima geração. Se essa taxa for muito alta pode acarretar em estagnação na busca, pois muitos indivíduos de uma geração serão automaticamente repassados para a geração seguinte. Isto reduzirá a

diversificação da população ao longo das gerações. Então as taxas mais baixas são as mais empregadas. Em muitos casos apenas o melhor indivíduo da população é automaticamente repassado para a geração seguinte.

3.4 Fluxo de execução dos Algoritmos Genéticos

Nesta seção será explicado o fluxograma básico dos Algoritmos Genéticos.

Inicialmente cria-se a população. Os cromossomos dos indivíduos presentes nessa primeira geração podem ser gerados aleatoriamente ou gerados nas regiões mais promissoras do espaço de busca. Essa segunda forma é possível quando se tem um conhecimento prévio sobre o espaço de busca do problema. Em seguida cada indivíduo é avaliado de acordo com a função de avaliação do problema para definir a aptidão. Após a fase de avaliação, testa-se um critério de parada, que é a condição na qual se considera que o algoritmo encontrou uma solução aceitável ou simplesmente quando se atingir a quantidade máxima de gerações estipuladas inicialmente. Caso o critério de parada tenha sido satisfeito então o melhor indivíduo é apresentado como a solução do problema e a execução é encerrada. Caso contrário, aplicam-se os operadores na população, em seguida partindo para a próxima geração (nova iteração do algoritmo). O algoritmo retorna à fase de avaliação e segue os passos já descritos. A Figura 14 mostra o fluxograma de execução do algoritmo.

Figura 14. Fluxograma básico de execução do Algoritmo Genético.

Capítulo 4

Modelo Proposto e Ferramenta

4.1 Modelo Proposto

No modelo proposto os indivíduos representam um conjunto de caixas colocadas sobre um palete. Para codificar esse conjunto e formar os cromossomos com o tamanho devido, é necessário obter a quantidade máxima de caixas que podem ser carregadas no palete. Informação que é facilmente encontrada dividindo a área do palete pela área da caixa. Uma caixa constitui um gene com dois números inteiros e um binário. Os dois inteiros são as coordenadas (x,y) da quina superior esquerda da caixa em relação à superfície do palete e o binário representa a orientação da caixa ("0" = horizontal e "1" = vertical) em relação à maior dimensão da caixa, o comprimento. A Figura 15 ilustra um cromossomo e o carregamento (caixas de medidas 4 x 2) gerado pela decodificação. As medidas do palete e das caixas, no modelo proposto, não possuem uma unidade de medida específica, pois não é relevante para o problema abordado. O que de fato é relevante é a proporção entre as medidas, por isso não é se estabelecer uma unidade de medida. Pelo mesmo motivo também não foi especificado uma unidade de área.

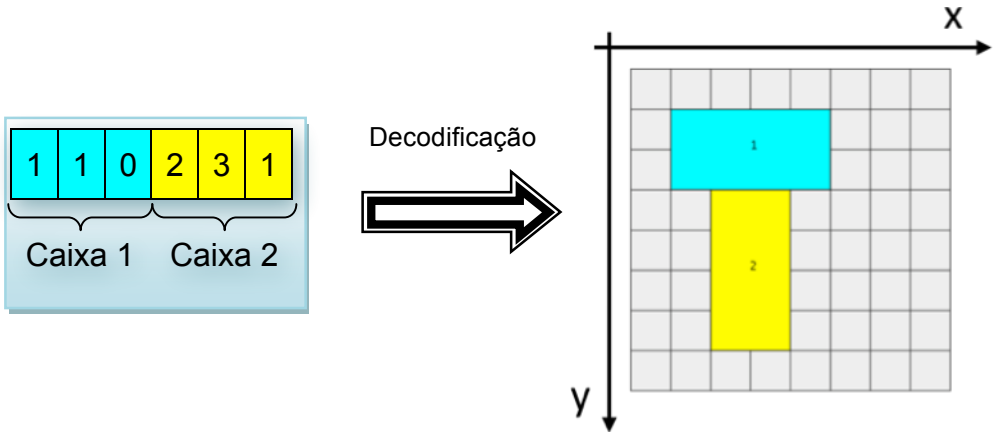

Figura 15. Cromossomo com dois genes (caixas) e carregamento gerado após a decodificação.

No modelo proposto, a função objetivo (1) é a razão entre o total de unidades de área com sobreposição de caixas (Sp) e o percentual de área que as caixas ocupam no palete (P_A).

$$F = \frac{Sp}{P_A} \tag{1}$$

O valor de Sp é calculado de acordo com o Algoritmo 3. O objetivo do problema é minimizar esta função, ou seja, os indivíduos mais aptos são os que apresentam menor valor quando avaliados pela função.

Algoritmo 3. Cálculo de sobreposição total (Sp).

1. *Decodifique o indivíduo gerando uma lista de caixas.*
2. *Para cada caixa na lista faça:*
   - 2.1. *Coloque a caixa no palete.*
   - 2.2. *Conte a quantidade de unidades de área que já estão ocupadas no local que a caixa foi posta (se há mais de uma caixa sob os pontos da caixa posta, então se deve contar tantas vezes quantas forem as caixas que ocupam aquele ponto).*
   - 2.3. *Incremente Sp com o valor encontrado no passo anterior.*
3. *Retorne Sp.*

4.1.1 Operadores

Nesta subseção será explicado como os operadores do Algoritmo Genético foram empregados para possibilitar a solução de problemas de carregamento de paletes. Foi necessário adequar os operadores e impor certas restrições para que os indivíduos não gerassem soluções inválidas para o problema. Ao longo deste tópico serão descritos tais adequações e restrições.

a) Mutação

Para sofrer mutação, um indivíduo precisa primeiramente ser escolhido. Há um valor percentual que indica a probabilidade de um indivíduo ser escolhido. Tal valor é único para toda a população e é definido pelo usuário do sistema. Por exemplo, se a

probabilidade de mutação é 5%, cada indivíduo tem 5% de chance de ser escolhido para sofrer mutação.

Da mesma forma que um indivíduo tem a probabilidade de ser escolhido para a mutação, um elemento do seu cromossomo tem uma probabilidade de ser alterado, denominada neste trabalho como probabilidade de mutação local. Assim como a probabilidade de mutação esse valor é único para toda a população e é definido pelo usuário. Um elemento que é escolhido para ser alterado será substituído por um valor aleatório obedecendo às restrições do problema. Uma dessas restrições está associada à posição do elemento no gene, logo o valor aleatório deve obedecer aos limites dos seguintes intervalos:

- Posição 1 (valor de x): deve estar no intervalo [0,C - 1], onde C é o comprimento do palete.
- Posição 2 (valor de y): deve estar no intervalo [0,L - 1], onde L é a largura do palete.
- Posição 3 (orientação da caixa): 0 ou 1 (“0” = horizontal e “1” = vertical).

Os intervalos das Posições 1 e 2 poderiam ser mais restritos de forma que não fosse possível colocar uma caixa fora do palete. No entanto, o limite de cada intervalo iria variar de acordo com a orientação da caixa, sendo necessária a verificação da orientação da caixa antes de limitar o intervalo. Então foi usado os intervalos descritos para facilitar a implementação do modelo proposto.

As restrições existentes para cada posição do gene são necessárias (mas não suficientes) para evitar que uma caixa fique fora do palete quando uma solução é gerada a partir da decodificação de um cromossomo. As caixas devem estar completamente postas no palete. Caso contrário a solução é inválida. Tal situação ocorre no caso demonstrado na Figura 16, a qual mostra um gene e sua decodificação.

Se a situação descrita na Figura 16 ocorrer, então será feita uma mutação no gene que representa a caixa no caso não permitido até que a caixa esteja completamente no palete. Essa mutação por gene também é realizada durante a criação da população inicial, já que os cromossomos dos indivíduos são gerados aleatoriamente e podem gerar soluções inválidas. Destacando que soluções com

sobreposição de caixas são consideradas válidas, pois a quantidade de sobreposição é necessária para avaliar a solução. No entanto, é possível optar por retirar as caixas sobrepostas na solução final, como será descrito na seção 4.2.

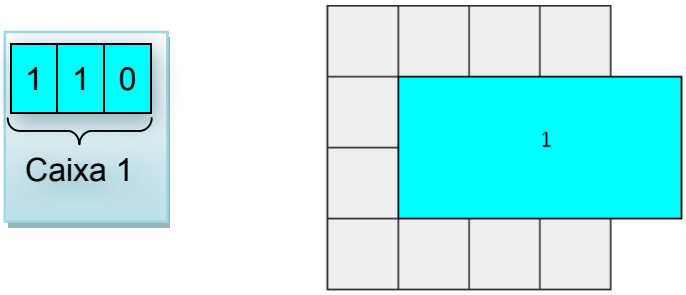

Figura 16. Caixa fora do palete. Caso não permitido.

b) Cruzamento

Foi utilizado o cruzamento de dois pontos. Essa operação é realizada da mesma forma que descrita na seção 3.2.1, mas mantendo as restrições, explicitadas no tópico a), para não gerar soluções inválidas.

c) Seleção

A seleção utiliza um critério simples. A quantidade de indivíduos da população (n: definida pelo usuário) deve ser a mesma ao final de cada geração. Sendo assim, se a população crescer devido à operação de cruzamento, então será necessário excluir alguns indivíduos até sobrar os n primeiros. A exclusão é feita após os indivíduos serem ordenados do mais apto para o menos apto, restando a cada geração os n melhores.

No Capítulo 5 será explicado o motivo de não ter sido utilizado a seleção por roleta.

d) Elitismo

A operação de elitismo consiste em manter determinado percentual de indivíduos sem sofrer mutação (E: definido pelo usuário). Logo, os E % melhores indivíduos da população deveriam não sofrer mutação, mas visando melhorar as respostas do algoritmo, foi implementada para tais indivíduos uma operação chamada mutação

guiada. Nesta operação a mutação só é feita no indivíduo se a aptidão dele melhorar, caso contrário ele continua com o cromossomo inalterado.

4.2 Ferramenta - SIP

A Figura 17 mostra a tela principal da ferramenta.

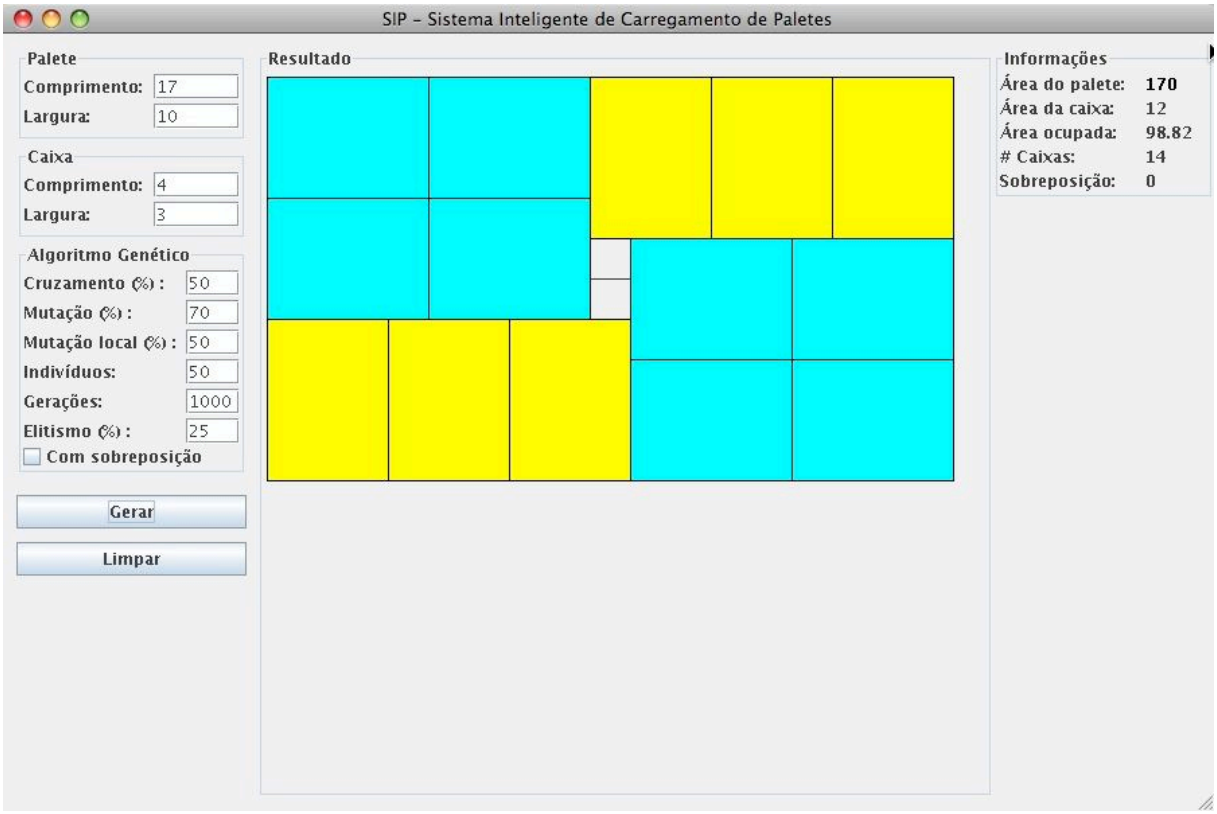

Figura 17. Tela principal do SIP.

Na Figura 17 os quadros *Palete* e *Caixa* contêm os campos *Comprimento* e *Largura*, nos quais o usuário irá especificar as medidas do palete e das caixas que serão postas sobre este (este trabalho trata somente de carregamentos com caixas idênticas, por isso existem campos para colocar medidas apenas de uma caixa). No quadro *Algoritmo Genético* estão os campos relativos aos parâmetros do algoritmo. No quadro central (*Resultado*) é mostrado o carregamento que o sistema gerou a partir dos dados de entrada. Na interface gráfica o tamanho dos quadrados que representam a unidade de área é inversamente proporcional ao comprimento do palete, isso foi feito para que o comprimento do palete sempre ocupe todo o comprimento do quadro *Resultado.* Os retângulos amarelos representam as caixas na posição vertical e os retângulos azuis representam as caixas na posição

horizontal. Adotou-se a diferenciação de cores de acordo com a posição da caixa para facilitar a visualização do carregamento. No quadro *Informações* são mostrados os dados do carregamento. O botão *Gerar* tem a função de executar o algoritmo para gerar o carregamento. O botão *Limpar* simplesmente apaga o carregamento gerado. Por fim, o *checkbox Com sobreposição* permite escolher se o sistema pode gerar carregamentos com sobreposição de caixas ou não. Essa opção pode ser útil para casos em que o usuário precisa visualizar a situação do carregamento em determinada geração. Para isso, basta marcar a opção *Com sobreposição* e colocar no campo *Gerações* a geração que ele deseja visualizar. Se essa opção estiver desmarcada o sistema irá retirar as caixas que se sobrepõem a outras. Caso a opção esteja marcada o sistema permite carregamentos com caixas sobrepostas. A área sobreposta é demarcada em vermelho e no quadro de *Informações* é mostrada a quantidade de medidas de área que está sobreposta. A Figura 18 exibe um carregamento com sobreposição de caixas.

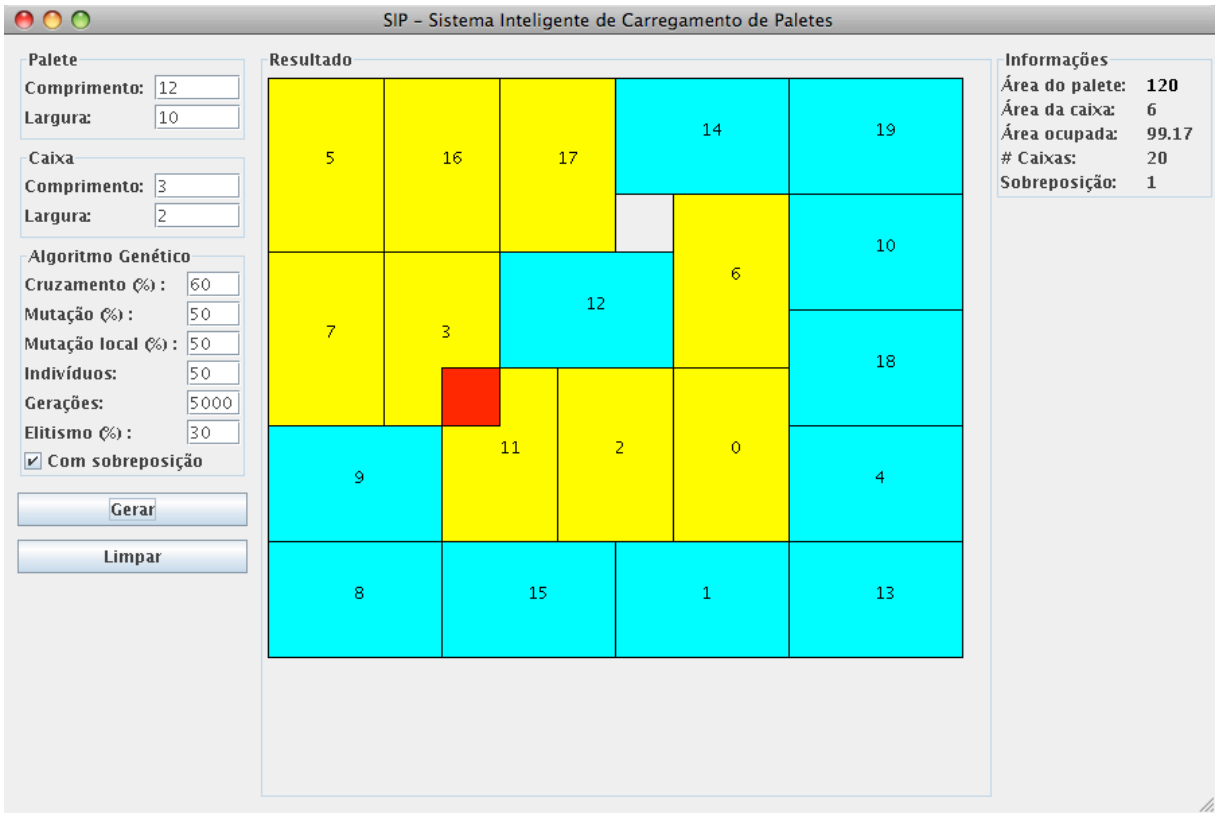

Figura 18. Carregamento com destaque na área que há caixas sobrepostas.

O SIP foi desenvolvido em linguagem Java [19] juntamente com a utilização da API (*Application Programming Interface*) gráfica Java 2D [20]. Ela foi necessária

para construir a visualização do carregamento presente no quadro *Resultado*, pois provê recursos que facilitam o desenho das caixas. O software está dividido em 3 camadas (ou módulos): Interface Gráfica, Carregamento e Algoritmos Genéticos. A Figura 19 mostra o diagrama de classes do sistema juntamente com a separação das camadas. No diagrama, as setas indicam dependência entre as classes, a classe para qual a seta aponta depende da classe da qual a seta é originada. Por exemplo, a classe *GuiController* depende da classe *PalletController*. Na modelagem do sistema, foi estabelecido que as classes de uma camada só devessem depender de classes da mesma camada ou da camada imediatamente inferior. Sendo assim, é possível alterar um módulo completamente sem que outro seja impactado, desde que os tipos das entradas e saídas dos módulos se mantenham. Esse fator facilita a evolução e a manutenção do sistema. Além das dependências, existem as associações entre classes, que são representadas por traços e possuem cardinalidade. Por exemplo, a classe *Pallet* tem uma associação de 1 para *n* com a classe *Box,* ou seja, na classe *Pallet* há uma lista de elementos da classe *Box*.

O módulo de Interface Gráfica possui a funcionalidade de prover interação do usuário com o sistema, o que é feito através dos campos de entrada e dos resultados gerados pela execução. Esse módulo é composto pelas classes:

- *MainPanel*: Painel principal que contém os componentes básicos para a inserção dos dados (campos de texto e informações).
- *ResultPanel*:Desenha o carregamento a partir dos dados resultantes da execução do algoritmo.
- *GuiController:* Controlador da interface. É responsável por captar os dados de entrada e repassá-los para camada inferior, bem como receber os dados resultantes e repassá-los para as outras duas classes deste módulo.

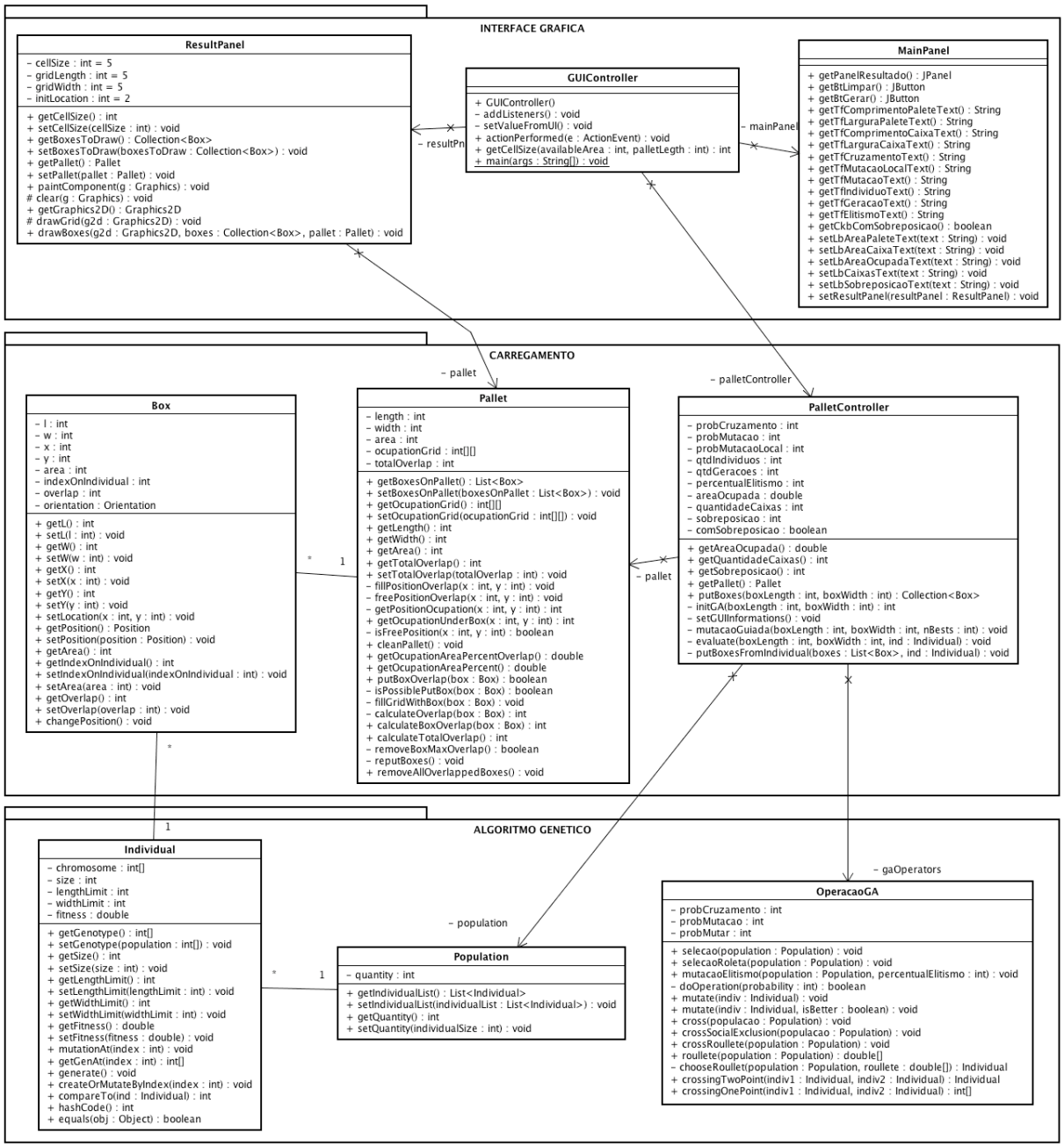

Figura 19. Diagrama de classes e módulos do SIP.

O módulo de Carregamento possui as classes:

- *Box*: Representa uma caixa no sistema. Possui atributos como: medidas (comprimento *l* e largura *w*); coordenadas cartesianas da extremidade superior esquerda da caixa; a orientação (horizontal ou vertical); e área.
- *Pallet*: Representa o palete no sistema. Possui atributos como: medidas (comprimento *l* e largura *w*); área; quantidade (unidades de

medida de área) total de sobreposição das caixas e a matriz de ocupação do palete. Essa classe executa ações importantes do sistema tais como cálculo da quantidade de sobreposição das caixas e colocação das caixas na matriz de ocupação.

- *PalletController*: Controlador do palete. É a classe que realiza a integração do problema de carregamento de paletes com os Algoritmos Genéticos. Nela, os indivíduos são decodificados para gerar as soluções do problema.

O módulo de Algoritmos Genéticos possui as classes necessárias para empregar essa técnica de computação evolucionária no problema de carregamento de paletes. As classes que compõem este módulo são:

- *Individual*: Representa um indivíduo. Possui atributos importantes para o funcionamento dos Algoritmos Genéticos, tais como: cromossomo e aptidão;
- *Population*: Encapsula o conjunto de indivíduos numa estrutura de dados do tipo lista;
- *OperacaoGA*: Possui os operadores do Algoritmo Genético: seleção, cruzamento, mutação e elitismo.

Capítulo 5

Experimentos e Resultados

5.1 Escolha dos Operadores

Inicialmente foram realizados experimentos preliminares para selecionar o tipo de cruzamento que seria utilizado no SIP. Foram testados três tipos de cruzamento: cruzamento por exclusão social, o modelo proposto descrito na seção 4.1.1 e cruzamento por roleta. Nestes experimentos a mutação e a seleção foram aquelas descritas na seção 4.1.1. Os dados utilizados foram:

- Medidas do palete: 12 x 10;
- Medidas da caixa: 4 x 3;
- Probabilidade de Cruzamento (somente para o método proposto): 50%;
- Probabilidade de Mutação: 50%;
- Probabilidade de Mutação Local: 50%;
- Percentual de Elitismo: 25%;
- Tamanho da população: 50 indivíduos;
- Número de gerações: 500.

Os valores dos parâmetros do Algoritmo Genético foram escolhidos empiricamente durante a execução de testes preliminares da ferramenta. Somente na realização dos experimentos o único critério de parada adotado foi o número de gerações, mas na ferramenta há também outro critério de parada, quando a sobreposição de caixas é 0. Para cada tipo de cruzamento foram realizadas 20 simulações. A partir das simulações extraiu-se média e desvio padrão do tempo de execução bem como a quantidade de vezes que o algoritmo convergiu para a solução ótima, ou seja, todas as caixas foram colocadas no palete sem sobreposição. Para as medidas do palete e da caixa escolhidas há três soluções ótimas, que são mostradas na Figura 20, Figura 21 e Figura 22.

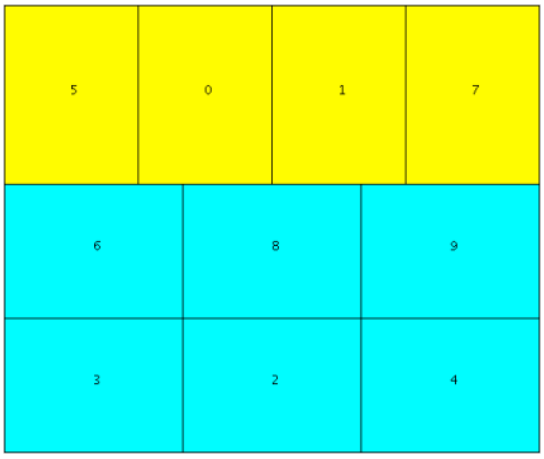

Figura 20. Solução ótima 1.

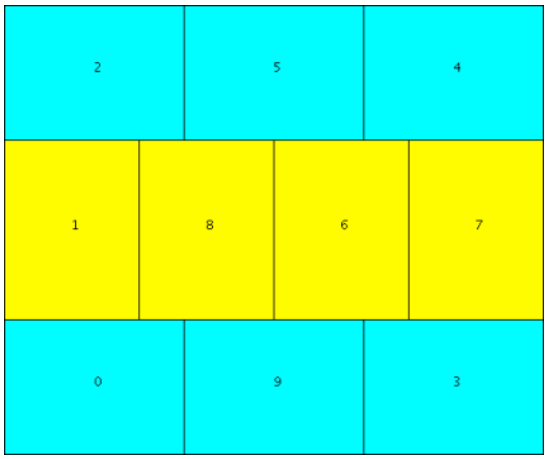

Figura 21. Solução ótima 2.

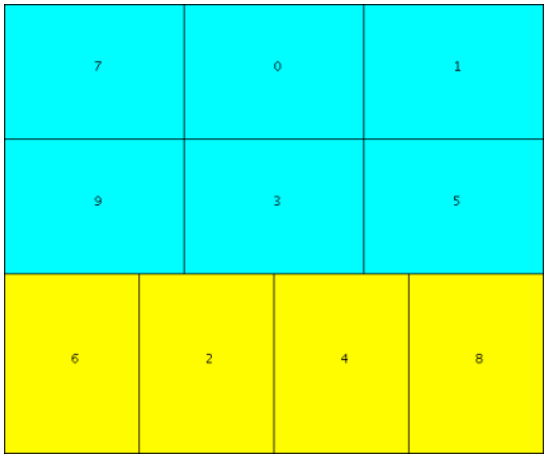

Figura 22. Solução ótima 3.

A Tabela 2 mostra os resultados das simulações.

Tabela 2. Resultados de simulações para escolha do tipo de cruzamento.

Cruzamento	Média (s)	Desvio Padrão	Quantidade de simulações que convergiu para a solução ótima
Exclusão social	0,77505	0,13846	3 (15%)
Proposto	3,51545	0,20947	13 (65%)
Roleta	22,5783	2,80646	0 (0%)

O método de exclusão social obteve a menor média de tempo e o menor desvio padrão dentre os três métodos, no entanto em apenas 15% das simulações o resultado ótimo foi alcançado. Já o método proposto obteve a média de tempo e desvio padrão um pouco maiores que o método de exclusão social, mas foi bastante superior no percentual de soluções ótimas, 65%. Por fim, o método de cruzamento por roleta não obteve resultado satisfatório, com sua média de tempo bem acima das outras duas. O desvio padrão também foi relativamente alto quando comparado com os demais, o que mostra maior variação no tempo de execução e por conseqüência menor precisão para a estimativa de tempo de execução de outras instâncias do problema. Além disso, o método de cruzamento por roleta não chegou à solução ótima em nenhuma das simulações. De posse desses dados, elegeu-se o método proposto para ser usado no SIP e para realizar outras análises do algoritmo, que serão descritas na seção 5.2. É importante ressaltar que para outros valores dos parâmetros do Algoritmo Genético os resultados poderão ser diferentes. Isso requer um estudo mais aprofundado da influência dos parâmetros sobre os resultados, o que é proposto como trabalho futuro.

O mesmo tipo de experimento realizado para escolha do tipo de cruzamento foi realizado para determinar o tipo de seleção que seria empregado, ou seja, os critérios para selecionar os indivíduos que irão compor a geração seguinte em cada iteração do algoritmo. Foram utilizados os mesmos dados do experimento anterior.

Os resultados estão na Tabela 3. Foi testada a seleção da seção 4.1.1 e a seleção por roleta, em ambos os casos utilizando o método de cruzamento proposto.

Tabela 3. Resultados de simulações para escolha do tipo de seleção.

Seleção	Média (s)	Desvio Padrão	Quantidade de simulações que convergiu para a solução ótima
Proposto	3,51545	0,20947	13 (65%)
Roleta	30,7654	0,12699	0 (0%)

Para o método de seleção proposto os dados permaneceram os mesmo do experimento para escolha do tipo de cruzamento, pois os operadores utilizados foram os mesmos (todos que foram descritos na seção 4.1.1). Já o método de seleção por roleta teve a média de tempo de execução substancialmente maior que o método proposto, além disso, o algoritmo não convergiu para a solução ótima em nenhuma das execuções. Com isso, definiu-se o método de seleção proposto para ser usado no SIP.

5.2 Resultados

Nos experimentos que serão mostrados nesta seção, foram usados os mesmos dados dos experimentos preliminares, com exceção do número de gerações e do tamanho da população. O primeiro foi aumentado para 2000 gerações e o segundo foram experimentados três valores (20, 40 e 60 indivíduos). Foram executadas 30 simulações para cada valor do tamanho da população. A partir disso, extrai-se o melhor resultado de cada geração. Com isso, calculou-se a média da ocupação da área do palete em cada geração. A Figura 23 mostra os resultados.

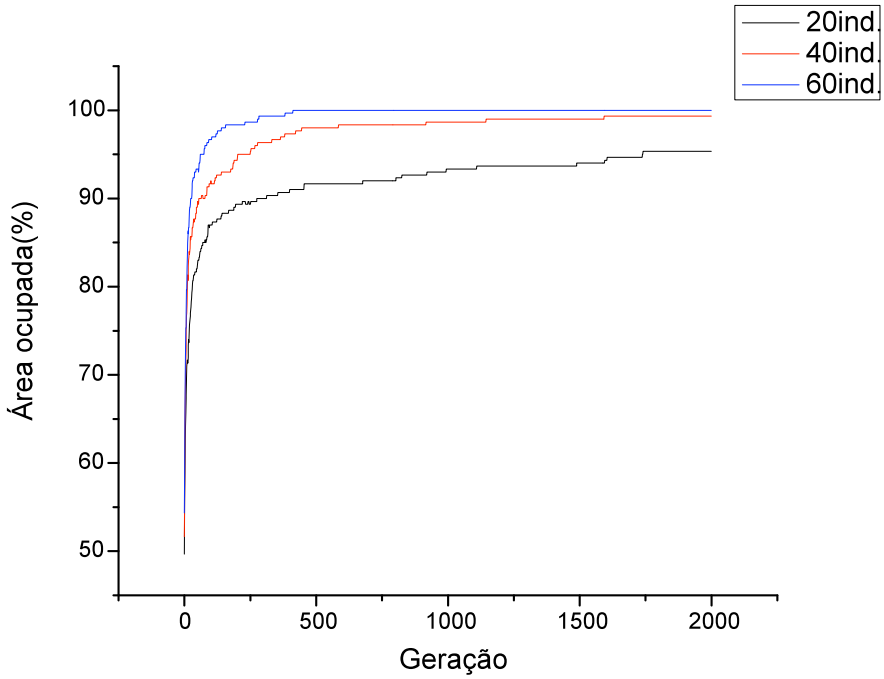

Figura 23. Evolução da ocupação de área do palete ao longo das gerações.

Com uma população de 60 indivíduos o algoritmo chegou a uma média de 100% de ocupação da área do palete a partir da geração 413. Por outro lado, nas simulações com 20 e 40 indivíduos, alcançou-se no máximo uma média de ocupação de área de 95,33% e 99,33%, respectivamente. Isso mostra que o tamanho da população tem forte influência sobre a convergência do algoritmo.

Para 20, 40 e 60 indivíduos o tempo de execução do algoritmo ao longo de 2000 gerações foi 2,51, 8,94 e 22,82 segundos, respectivamente. No caso de 60 indivíduos o tempo de execução até a geração 413 (geração que houve a convergência) foi de 4,81 segundos. Logo, o aumento no tempo de execução do algoritmo pode ser compensado pela aceleração na convergência para a solução ótima. Esses tempos de execução são os mesmos para a análise de sobreposição mostrada na Figura 24, pois as simulações retornam tanto a área ocupada quanto a sobreposição. Logo, ambos os dados foram extraídos nas mesmas simulações.

Na Figura 23 é importante destacar a alta aceleração na convergência nas primeiras gerações. Da geração 1 até a geração 50 a ocupação da área aumentou da seguinte forma para cada valor analisado:

- 20 indivíduos: de 49.67% até 82,33%;
- 40 indivíduos: de 51.67% até 89,67%;
- 60 indivíduos: de 54.33% até 93,33%.

Essa aceleração de convergência nas primeiras gerações mostra que na fase inicial a busca em amplitude leva o algoritmo rapidamente para as regiões mais promissoras do espaço de busca. Ao longo das gerações a velocidade de convergência vai reduzindo, isso devido à redução da busca amplitude para realização da busca em profundidade. Em outras palavras, o algoritmo faz uma varredura no espaço de busca e segue gradativamente limitando a busca nas regiões mais promissoras fazendo uma busca em profundidade, visando encontrar soluções mais refinadas que as boas soluções já encontradas.

Além das análises de convergência com relação à ocupação da área do palete com as caixas, fez-se uma análise em relação à sobreposição das caixas, valor medido em unidades de área. Utilizou-se a mesma metodologia de experimentos que foi usada nas análises de ocupação de área do palete. Os resultados dos experimentos estão expostos na Figura 24.

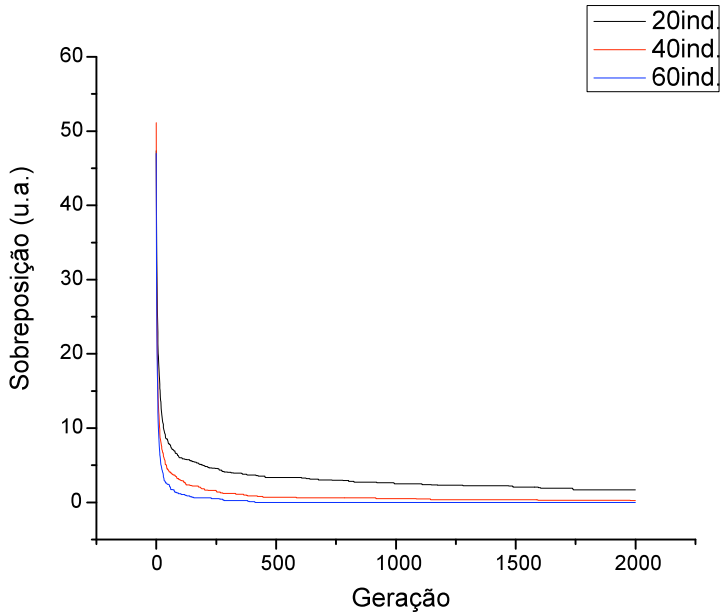

Figura 24. Gráfico de redução de sobreposição de caixas ao longo das gerações.

Como explicitado na Figura 24, com o passar das gerações a quantidade de sobreposições tende a 0. Este é justamente o objetivo do algoritmo, minimizar a quantidade de sobreposições e por conseqüência aumentar a ocupação da área do palete, já que as caixas ficam mais distribuídas sobre o palete. Apesar de neste trabalho uma solução com sobreposição diferente de 0 ser considerado válida (por ser um dos critérios de avaliação das soluções), para problemas reais esse tipo de solução não é válida, logo as caixas sobrepostas são desconsideradas. Sabendo disso, o único conjunto de simulações que pode ser considerado totalmente válido na prática é o de 60 indivíduos, pois atingiu 0 de média de sobreposição em 30 simulações ao longo de 2000 gerações. Os demais conjuntos de simulações, com 20 e 40 indivíduos, atingiram uma média de sobreposição de 1,7 e 0,23 respectivamente, ao longo das 2000 gerações.

5.3 Exemplos de soluções encontradas pelo SIP

Nesta seção serão mostradas algumas soluções encontradas pelo SIP com outras medidas de paletes e caixas. Na Figura 25, Figura 26 e Figura 27 mostram alguns casos em que o SIP encontrou a solução ótima do problema.

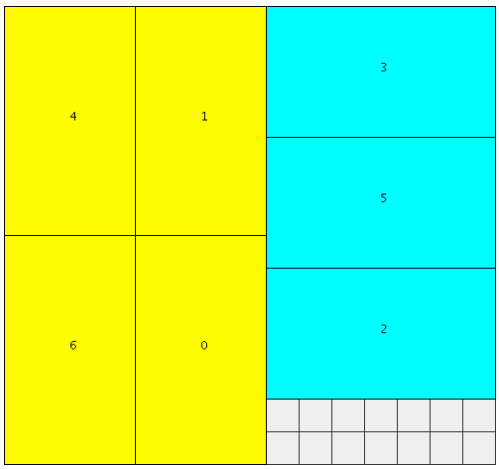

Figura 25. Solução com o palete (15 x 14) e caixa (7 x 4).

Na Figura 25 os resultados foram:

- Percentual de área ocupada do palete: 93,33%.
- Quantidade de caixas no carregamento: 7 caixas.

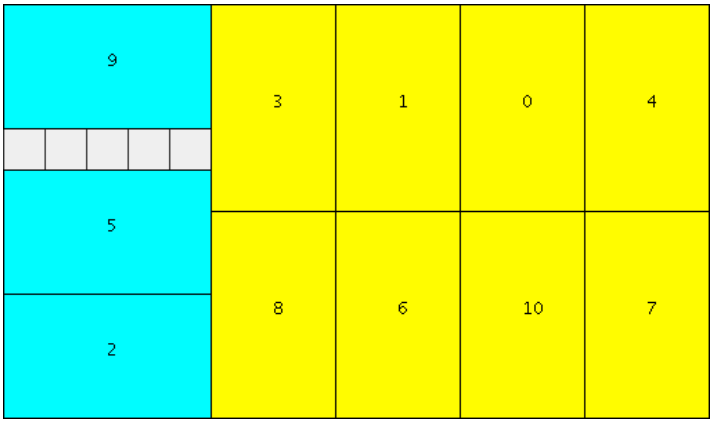

Figura 26. Solução com o palete (17 x 10) e caixa (5 x 3).

Na Figura 26 os resultados foram:

- Percentual de área ocupada do palete: 97,06%.
- Quantidade de caixas no carregamento: 11 caixas.

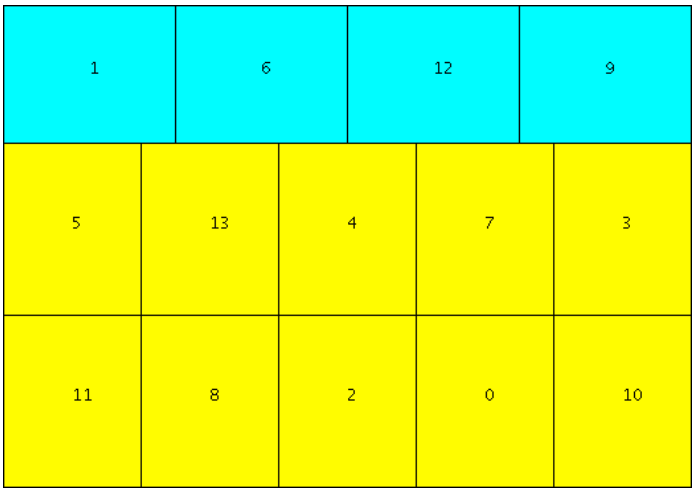

Figura 27. Solução com o palete (20 x 14) e caixa (5 x 4).

Na Figura 27 os resultados foram:

- Percentual de área ocupada do palete: 100%.
- Quantidade de caixas no carregamento: 14 caixas.

Capítulo 6

Conclusão e Trabalhos Futuros

Neste trabalho, foi desenvolvida uma nova técnica para resolução de problema de carregamento de paletes. Essa técnica incorporou o uso de Algoritmos Genéticos, o qual foi adaptado para o problema alvo deste trabalho. As principais adaptações foram na formação do cromossomo dos indivíduos e nos operadores, cujas adaptações foram feitas para atender às restrições do problema. Além das adaptações, foi proposto um novo tipo de operador de seleção e de cruzamento, que obtiveram, para este tipo de problema, um desempenho superior aos outros tipos de operadores testados. Além disso, na análise de resultados ficou evidente que a quantidade de indivíduos da população tem influência direta na convergência do algoritmo.

Desenvolveu-se, também, uma ferramenta computacional para resolução de problemas de carregamento de paletes, o SIP. A ferramenta chegou à solução ótima em problemas com carregamentos mais simples, de até 15 caixas. Em alguns experimentos preliminares notou-se que para problemas mais complexos não se conseguiu chegar à solução ótima e além do tempo de simulação aumentar substancialmente nesses casos, em carregamentos com mais de 100 caixas por exemplo. É necessária uma análise mais detalhada do tempo de simulação nesses casos para testar a escalabilidade do método proposto.

O SIP ainda não apresenta todos os requisitos necessários ao ponto de poder ser comercializado ou distribuído como software livre. No entanto, é uma ferramenta que pode ser utilizada para auxiliar no estudo de outras técnicas de inteligência computacional em problema de carregamento de paletes. É possível, com certos ajustes, incorporar outra técnica de inteligência computacional no sistema, pois sua modularização facilita esse processo.

6.1 Contribuições

Este trabalho pode ser um ponto de partida para se desenvolver uma nova linha de pesquisa no DSC (Departamento de Sistemas e Computação) em problemas de carregamento de palete, além de outros problemas relacionados, tais como: carregamento de contêineres [4], problemas de corte e empacotamento [21], entre outros problemas da área de logística.

Foi criada uma a ferramenta computacional, chamada SIP que pode futuramente gerar um produto que auxilie as transportadoras e operadores logísticos no processo de paletização. Isso contribuirá para a redução de custos envolvida neste processo, bem como em transporte e armazenagem de mercadorias.

6.2 Dificuldades Encontradas

Uma das principais dificuldades encontradas neste trabalho foi a explicação, em certos pontos, superficial de alguns artigos científicos estudados. Alguns detalhes eram fundamentais para a implementação de outros algoritmos, os quais seriam comparados com o método proposto. Além disso, não haveria tempo hábil para implementá-los e realizar os comparativos.

6.3 Trabalhos Futuros

Neste trabalho, foi proposto um método de resolução de problema de carregamento de palete utilizando Algoritmos Genéticos. Sendo assim, propõe-se futuramente realizar uma análise mais detalhada da influência dos operadores e parâmetros envolvidos para que se possa melhorar o desempenho da abordagem proposta.

Pode ser feito também uma análise do tempo de execução do algoritmo e como a quantidade de indivíduos influencia nesse tempo.

Aumentar as restrições das soluções válidas, ou seja, criar mais critérios para determinar a validade de uma solução. Isso poderá reduzir o espaço de busca e direcionar a busca para as melhores soluções, além de poder reduzir o tempo de execução.

Criar outros tipos de operadores no Algoritmo Genético. Por exemplo, um operador de refinamento das soluções. Ele efetuaria alterações nos indivíduos que corresponderiam a movimentos das caixas na direção das extremidades do palete. Isso pode abrir um espaço entre as caixas de maneira que possam caber mais caixas no palete.

Propor outra equação de avaliação dos indivíduos.

Expandir o algoritmo para criar carregamento de várias camadas considerando a estabilidade do conjunto de caixas [9].

Empregar outras técnicas de inteligência computacional para a resolução do problema abordado, por exemplo, PSO (*Particle Swarm Optimization*) [22].

Bibliografia

[1] RAM, B. The pallet loading problem: A survey, International Journal of Production Economics, p. 217-225, 1992.

[2] HODGSON, T. A combined approach to the pallet loading problem, IIE Transactions, v. 14 (3), p. 176-182, 1982.

[3] MASCARENHAS, W. F. Two aspects of the pallet loading problem, Electronic Notes in Discrete Mathematics, v. 19, p. 381–387, 2005.

[4] GEORGE, J.A.; ROBINSON, D.F. A heuristic for packing boxes into a container. Computers and Operational Research, v. 7, p. 147-156, 1980.

[5] YAMASSAKI, C.; PUREZA, V. Um refinamento do algoritmo tabu de Dowsland para o problema de carregamento de paletes do produtor, Relatório Técnico, Departamento de Engenharia de Produção, UFSCar, 2001.

[6] PUREZA, V.; MORABITO, R. Some experiments with a simple tabu search algorithm for the manufacturer's pallet loading problem, Computers & Operations Research, v. 33, p. 804 – 819, 2006.

[7] DOWSLAND, K. Some experiments with simulated annealing techniques for packing problems. European Journal of Operational Research, vol. 68, p. 389-399, 1993.

[8] HERBERT, A.; DOWSLAND, K. A family of genetic algorithms for the pallet loading problem, Annals of Operations Research, v. 63, p. 415-436, 1996.

[9] JAAKKOLA, J.; LEIPÄLÄ T.; NEVALAINEN O. On the Stability of Pallet Loading Layouts, University of Turku, Department of Mathematical Sciences and TUCS, 20014 Turku, Finland.

[10] DYCKHOFF, H. A. A typology of cutting and packing problems. European Journal of Operational Research, v. 44, p. 145-159, 1990.

[11] MORABITO, R.; MORALES, S. A simple and effective recursive procedure for the manufacturer's pallet loading problem. Journal of the Operational Research Society, v. 49, p. 819-828, 1998.

[12] HOLLAND, J. H. Adaptation in Natural and Artificial Systems, Ann Arbor, MI: The University of Michigan Press, 1975.

[13] DARWIN, C. On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life. Editora: John Murray, Londres, 1859.

[14] PAPADIMITRIOU, C. H.; STEIGLITZ, K. Combinatorial Optimization: Algorithms and Complexity. Estados Unidos, Edição: 1, Editora: DOVER SCIENCE, 1998.

[15] SALLABI, O. M.; EL-HADDAD, Y. An Improved Genetic Algorithm to Solve Traveling Salesman Problem. Proceedings of World academy of Science, Engineering and Technology, Roma, Itália, v. 40, 2009.

[16] LIENIG, J.; COHOON, J. P. Genetic algorithms applied to the physical design of VLSI circuits: A survey. International Conference on Evolutionary Computation, Berlin, vol. 1141, p. 839-848, 1996.

[17] PINTO, A.R.; BORGES, P. S. S. Comparação de métodos de seleção de reprodutores alternativos com o método da roleta. INE-CTC – Universidade Federal de Santa Catarina, Florianópolis.

[18] SRINIVAS, M.; PATNAIK, L. M. Genetic algorithms: A survey. IEEE Computer, vol. 27, p. 17–26, 1994.

[19] Java Technology: http://java.sun.com. Acesso em 23 de maio de 2009.

[20] Java 2D API: http://java.sun.com/products/java-media/2D. Acesso em 23 de maio de 2009.

[21] ALVES, C. M. M. Cutting and Packing: Problems, Models and Exact Algorithms. Braga, 2005. 175 p. Dissertação (Doutorado em Engenharia de Produção e Sistemas) - Departamento de Produção e Sistemas da Escola de Engenharia, Universidade do Minho.

[22] KENNEDY, J.; EBERHART, R. Particle swarm optimization, Neural Networks IEEE International Conference, Austrália, v. 4, p. 1942-1948, 1995