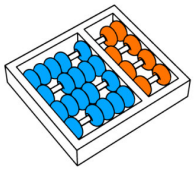
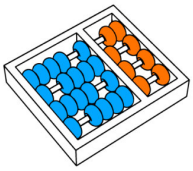


Tiago Silveira

"Problemas de Empacotamento com Itens Irregulares: Heurísticas e Avaliação de Construtores de NFP"

CAMPINAS
2013



Universidade Estadual de Campinas
Instituto de Computação

Tiago Silveira

"Problemas de Empacotamento com Itens Irregulares: Heurísticas e Avaliação de Construtores de NFP"

Orientador(a): **Prof. Dr. Eduardo Candido Xavier**

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Computação da Universidade Estadual de Campinas para obtenção do título de Mestre em Ciência da Computação.

Este exemplar corresponde à versão final da Dissertação defendida por Tiago Silveira, sob orientação de Prof. Dr. Eduardo Candido Xavier.

Assinatura do Orientador(a)

CAMPINAS
2013

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

Si39p

Silveira, Tiago, 1987-
Problemas de empacotamento com itens irregulares : heurísticas e avaliação de construtores de NFP / Tiago Silveira. – Campinas, SP : [s.n.], 2013.

Orientador: Eduardo Candido Xavier.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

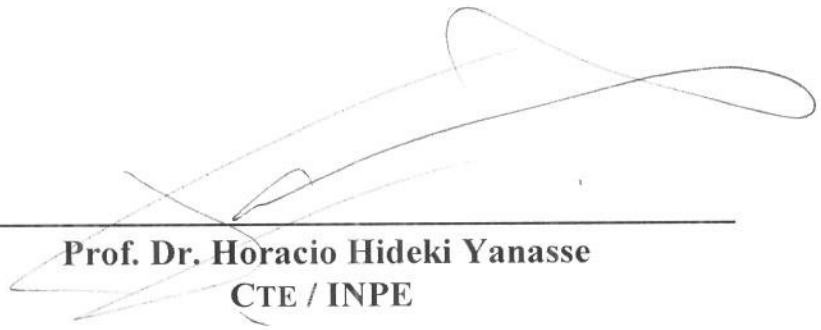
1. Problema de empacotamento. 2. Otimização combinatória. 3. Geometria computacional. 4. Programação heurística. 5. Algoritmos genéticos. I. Xavier, Eduardo Candido,1979-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Irregular packing problems : heuristics and evaluation of NFP constructors
Palavras-chave em inglês:
Packing problem
Combinatorial optimization
Computational geometry
Heuristic programming
Genetic algorithms
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Eduardo Candido Xavier [Orientador]
Horacio Hideki Yanasse
Flávio Keidi Miyazawa
Data de defesa: 29-07-2013
Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 29 de Julho de 2013, pela Banca examinadora composta pelos Professores Doutores:



Prof. Dr. Horacio Hideki Yanasse
CTE / INPE

Prof. Dr. Flávio Keidi Miyazawa
IC / UNICAMP

Prof. Dr. Eduardo Cândido Xavier
IC / UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Problemas de Empacotamento com Itens Irregulares: Heurísticas e Avaliação de Construtores de NFP

Tiago Silveira[1]

29 de julho de 2013

Banca Examinadora:

- Prof. Dr. Eduardo Candido Xavier (*Orientador*)
- Prof. Dr. Horacio Hideki Yanasse
  Instituto de Ciência e Tecnologia - UNIFESP
- Prof. Dr. Flávio Keidi Miyazawa
  Instituto de Computação - UNICAMP
- Prof. Dr. Orlando Lee
  Instituto de Computação - UNICAMP (suplente)
- Prof. Dr. André Luis Vignatti
  Departamento de Informática - UFPR (suplente)

[1]Suporte financeiro: Bolsa da CAPES (processo 01-P-04388-2010) 2010–2011

Abstract

Packing problems appears in various scenarios of the industry and techniques developed for their treatment are of great use. Among the problems that can be modeled as packing problems we have the optimization of layouts of pieces of wood, textiles products, paper, metal sheets, plastic, glass etc..

This paper present a heuristic based in the meta-heuristic genetic algorithm for the Strip Packing, Knapsack and Bin Packing problems. In all there problems, we consider two-dimensional irregular items to be packed in a bin of rectangular shape. In general, the proposed heuristic takes a list of items to be packed and builds the solution to the problem, taking into account the order of items in the list and some geometric features of the items already packed in a partial solution. We apply a local search on the solution using three operators that change the order of items in the list, allowing the generation of new solutions.

As an extension, we adapted the strip packing version of the heuristic to compact bins while packing new items when solving the Bin Packing and Knapsack problems.

We conduct computational experiments to compare the proposed heuristics with others of the literature. Our heuristic for the Knapsack problem has obtained better results than previous heuristics. For the other problems, our heuristic obtained competitive results when compared to the ones of the literature.

In addition, we implemented and made a performance analysis of three algorithms that generate the geometric structures called "No-Fit Polygons" (NFPs), which can be used for the verification of overlap between items when resolving irregular packing problems.

Resumo

Os problemas de empacotamento estão presentes em vários cenários da indústria e as técnicas desenvolvidas para o seu tratamento são de grande utilidade. Entre os problemas que podem ser modelados como problemas de empacotamento estão a otimização de leiautes em peças de madeira, de produtos têxteis, de folhas de papel e de metal, de plástico e de vidro etc.

Este trabalho apresenta uma nova heurística baseada na meta-heurística Algoritmo Genético para os problemas de empacotamento *Strip Packing*, *Knapsack* e *Bin Packing*, considerando a variação bidimensional com itens irregulares. De uma forma geral, a heurística proposta toma uma lista de itens a ser empacotada e constrói a solução para o problema, levando em conta a ordem dos itens na lista e algumas características geométricas do item empacotado com a solução parcial. Aplicamos uma busca local na solução utilizando três operadores que alteram a ordem dos itens na lista, possibilitando a geração de novas soluções.

Como extensão, adaptamos a heurística proposta para os problemas *Bin Packing* e *Knapsack* com o algoritmo proposto para o problema *Strip Packing*, de forma que recipientes específicos possam ser compactados durante a construção da solução.

Realizamos experimentos computacionais para comparar as heurísticas propostas com outras da literatura. Nossa heurística para o problema *Knapsack* obteve melhores resultados que os de outras heurísticas. Para os demais problemas, nossa heurística obteve resultados competitivos quando comparado com outros da literatura.

Além disso, implementamos e fizemos uma análise de desempenho de três algoritmos que geram as estruturas geométricas chamadas "*No-Fit Polygons*" (NFPs), que podem ser utilizadas na verificação de sobreposição entre itens durante a resolução de problemas de empacotamento com itens irregulares.

Aos meus pais, Cirene e Mauro, e irmãos, Gabriel e Gustavo.

Agradecimentos

Primeiramente, a Deus, pela graça oferecida para a conclusão deste trabalho.

À minha mãe, Cirene, ao meu pai, Mauro, ao meu 'irmãozinho', Gabriel e ao meu 'irmãozão', Gustavo, pela presença, amor e carinho, fundamentais à cada dia.

Ao Instituto de Computação da Universidade Estadual de Campinas, pela oportunidade oferecida, e que hoje muito me alegra esta decisão tomada.

Ao secretário do IC/UNICAMP, Daniel Capeleto, pela disposição em me ajudar desde os primeiros dias, principalmente nos momentos em que mais necessitei de ajuda.

Ao Professor Dr. Eduardo Candido Xavier, orientador deste trabalho, que muitas vezes foi além, sendo conselheiro, amigo e motivador, com suas ideias sinceras, sensatas e justas. Agradeço, também, por sua dedicação e empenho, pelos conhecimentos transmitidos, pela paciência durante as nossas reuniões semanais (principalmente) e pela confiança em mim depositada para a realização deste trabalho.

Ao professor Pedro Jussieu de Rezende, por seus conhecimentos transmitidos e pelas atitudes presenciadas em sala de aula, que me motivam a sempre ser uma pessoa melhor. Agradeço pelo exemplo de excelência didática, pela preocupação constante em melhorar o meu aprendizado (inúmeras solicitações de ajuda prontamente atendidas) e pelos conselhos em momentos de difíceis decisões, que muito me ajudaram.

Ao amigo, colega de trabalho e sempre professor Dr. Humberto César Brandão de Oliveira, pelo estímulo e apoio incondicional. Agradeço sua longeva paciência e enorme vontade de sempre me ajudar a crescer profissionalmente.

Aos demais amigos de trabalho da UNIFAL-MG que sempre me apoiaram. Agradeço a Luiz Eduardo, Paulo Bressan, Rodrigo Pagliares, Flavio Gonzaga, Ricardo Salgado.

Aos grandes amigos, que me ajudaram diretamente a concluir esta fase: Jair Trapé, Carlos Eduardo, Fernando Henrique, Carol(ine) Letizio, Lucas Oliveira, Guilherme Santos. Também, aos que me ajudaram indiretamente: Mara Ávila, Elana Freire, Tati(ane) Fernandes, Filipe Costa, Max Moreira, Jefferson Moisés, Kaio Karam, Vinicius Novaes, Guilherme Kunigami.

E, aos demais que não tenha me lembrado nesse momento, minha eterna gratidão!

"Dificuldades são oportunidades."
Joseph Murray

Sumário

Capítulo 1

Introdução

Este capítulo faz a apresentação dos problemas tratados neste trabalho. Além disso, são apresentadas as contribuições obtidas com o seu desenvolvimento.

Os problemas de corte e empacotamento estão presentes em vários cenários da indústria e as técnicas desenvolvidas para o seu tratamento são de grande utilidade.

De uma maneira informal, problemas de corte referem-se a divisão de unidades maiores (objetos) em unidades menores (itens). Já os problemas de empacotamento referem-se ao agrupamento de unidades menores (itens) em unidades maiores (objetos ou recipientes). Em geral, o objetivo nestes problemas é maximizar a utilização do espaço no qual os itens estarão contidos, evitando, assim, desperdício de materiais (matérias-primas).

Dada uma instância de um problema de corte e empacotamento a se resolver, muitas vezes não é necessário obter uma solução de forma imediata. No entanto, construir soluções manualmente poderá dispender muito tempo, de modo que a demora para se gerar um leiaute[1] aceitável torne o processo inviável. Dessa forma, a solução algorítmica torna-se necessária para grande parte das indústrias, a fim de serem produzidos leiautes com considerável fator de aproveitamento.

Embora a redução dos custos da produção industrial estimule o desenvolvimento de novas estratégias de geração de solução, a criação destas para se obter leiautes admissíveis nem sempre é simples, e, dependendo da geometria dos itens utilizados e das restrições impostas ao processo, pode se tornar ainda mais complexa. Uma das consequências disso é o aumento, também, da complexidade dos algoritmos geométricos utilizados para se produzir uma solução com determinado grau de precisão e qualidade.

[1] Modo de distribuição e arranjo dos itens em um determinado recipiente.

1.1 Aplicações práticas na indústria

Os problemas de corte e empacotamento são encontrados em diferentes tipos de indústrias. Entre os problemas que podem ser modelados como de corte e empacotamento estão a otimização de leiautes em peças de madeira, de produtos têxteis, de folhas de papel e de metal, de plástico e de vidro etc. Apesar desta diversidade, a resolução destes problemas está vinculada, na maioria das vezes, à maximização da utilização de algum material, pois a redução de desperdício, mesmo que pequena, atrelada à produção em massa das indústrias reduzem, consideravelmente, os seus custos de produção, deixando-as cada vez mais competitiva no mercado (Li e Milenkovic [30]).

A seguir, destacamos as características que fazem parte do dia-a-dia de algumas indústrias do ramo de corte e empacotamento (Hopper [25]).

1.1.1 Indústria de peças de metal

Geralmente, este tipo de indústria trabalha tanto com itens regulares (conjunto formado exclusivamente por itens com formatos de quadrados ou retângulos ou triângulos ou círculos) quanto com irregulares (conjunto formado por itens de qualquer formato). Apesar dessa abrangência, existem características e restrições impostas ao problema que as separam de outras indústrias. Por exemplo, suas matérias-primas podem ser representadas por peças de metal de dimensão fixa ou por pequenos rolos de metal de largura fixa. A condição de se trabalhar com objetos de dimensão fixa acaba, muitas vezes, dificultando a resolução do problema, devido as sobras de materiais exigirem que muitos leiautes diferentes sejam produzidos para um mesmo conjunto de itens. Outra característica é a constituição da matéria-prima, que pode apresentar uma não homogeneidade ou determinada orientação em seu aspecto físico, que irá influenciar diretamente a disposição dos itens da solução.

Um fator muito importante quando se trata um problema de corte e empacotamento em indústrias desse ramo é a técnica usada para cortar o material. Dada a tecnologia para realização do corte, todo o leiaute poderá sofrer impacto, visto que a espessura deste corte (ou seja, a quantidade descartada do material pela técnica de corte para separar os itens de um leiaute) tem influência sobre cada um dos itens dispostos sobre o leiaute construído.

Por fim, a ordem de geração dos itens pode ser importante dentro de um processo de produção em massa. Por exemplo, restrições de peso, tamanho do item ou outras características geométricas destes podem exigir que certos itens sejam produzidos em uma determinada ordem, a fim de agilizar a logística para obtenção do produto final.

1.1.2 Indústria de produtos têxteis

Por se ter uma fácil manipulação de suas matérias-primas, as indústrias têxteis usualmente dispõem de grandes rolos de materiais em sua produção. Com isso, a geração de novos leiautes pode ser facilitada, visto que o leiaute resultante irá utilizar apenas uma pequena parte do recipiente disponível. A maior dificuldade encontrada por tais indústrias está ligada à constituição física da matéria-prima. Como o tecido utilizado na produção das peças têm, geralmente, uma direção de fabricação (referente à direção do entrelaçamento dos fios dos tecidos), tem-se, também, um padrão de produção dos itens, restringindo a livre rotação destes no material. Dessa forma, itens individuais são produzidos considerando somente duas rotações sobre o leiaute – 0 e 180 graus –, sendo uma restrição que, na maioria dos casos, diminui o aproveitamento do leiaute final.

1.1.3 Indústria de produtos de couro

Um dos problemas práticos de corte e empacotamento mais difíceis de se resolver estão relacionados aos das indústrias de produtos de couro. Nesse caso, a constituição física das matérias-primas utilizadas impõem uma restrição grande ao processo, vista a possibilidade de enorme irregularidade tanto do objeto –por ser um produto natural– quanto dos próprios itens. A análise da qualidade esperada para os itens produzidos (que envolve cor, constituição física do material etc.) é outro ponto determinante na criação de um leiaute, de modo que deve ser considerado por qualquer estratégia de geração de solução, a fim de se conseguir utilizar a maior área possível dos materiais disponíveis.

1.1.4 Indústrias que utilizam itens regulares

Nos casos de se ter apenas itens regulares, o processo de criação dos leiautes pode ser um pouco simplificado. Aqui, os leiautes são produzidos dispondo os itens seguindo padrões de cortes. Um padrão de corte "guilhotinado", no qual os itens podem ser obtidos por cortes retos sobre o objeto restante, acaba inserindo algumas restrições ao problema devido as características exigidas do leiaute final produzido. Porém, as situações em que temos um padrão de corte "não guilhotinado", nas quais a condição anterior não necessariamente é seguida, acaba dando uma maior liberdade na resolução do problema. As principais indústrias deste segmento são as de peças de madeira, de papel, de vidro e de determinados setores logísticos.

1.2 Abordagens para geração de soluções

Há uma grande variedade de abordagens para geração de solução para os problemas de corte e empacotamento. As abordagens exatas para resolução destes são aplicáveis em alguns casos, sendo, na maior parte das vezes, aqueles em que se considera uma variação regular do problema; o uso da programação linear é bastante difundida nestes casos. No entanto, tais abordagens se restringem a instâncias pequenas do problema, sendo, geralmente, não compatível com a exigência das aplicações práticas.

Visto que obter a solução ótima é interessante, porém inviável em grande parte dos casos, os algoritmos aproximados surgem como boas alternativas. Com estas estratégias, o tempo gasto no processo de geração de soluções é consideravelmente menor, sendo que, geralmente, a qualidade do resultado obtido não é muito prejudicada. A principal desvantagem em adotar esta abordagem, semelhante às abordagens exatas, é a dificuldade de se elaborar heurísticas de resolução para determinadas variações destes problemas (variação irregular, por exemplo) e, quando desenvolvida, a estratégia proposta resolve com qualidade apenas o caso específico para a qual foi modelada.

Buscando fornecer uma abordagem mais geral, a aplicação de heurísticas na resolução destes problemas vem sendo cada vez mais difundida. Um dos principais motivos se refere à sua rápida execução, sendo que os resultados obtidos tendem a apresentar uma boa qualidade. Dentro da área de heurísticas, temos as chamadas *meta-heurísticas*, que são heurísticas genéricas para resolução de problemas de otimização. O uso das meta-heurísticas faz com que a resolução destes problemas torne-se mais flexível, pelo fato de seu modelo de otimização ser baseado em simples princípios de busca, em vez de regras específicas a serem seguidas, mesmo quando é necessário a adição de várias restrições, impostas por problemas reais. Exemplos de meta-heurísticas aplicadas em problemas de corte e empacotamento são: *Simulated Annealing*, Busca Tabu, *Hill Climbing*, Otimização por Enxame de Partículas, Algoritmo Genético (apresentado no capítulo 5) etc..

1.3 Objetivos

Como descrito por Dyckhoff [16], tanto o problema de empacotamento quanto o problema de corte apresentam uma estrutura lógica equivalente, o que lhes possibilitam formulações semelhantes, proporcionando a resolução pelas mesmas estratégias de solução. Por esse motivo, o foco deste trabalho foi direcionado apenas aos problemas de empacotamento, a fim de se produzir novas estratégias para geração de soluções (seção 5.3). Além disso, foi adotada a versão bidimensional do problema que considera itens irregulares, de forma que foi dada uma atenção especial à geometria do problema referente às estruturas utilizadas para se decidir sobreposição entre itens e para se encontrar posições viáveis de

agrupamento dos itens durante o processo de empacotamento. Tais estruturas referem-se aos "*No-Fit Polygons*" (seção 4.3).

1.4 Contribuições

São duas as principais contribuições deste trabalho. A primeira é a implementação e comparação de três algoritmos da literatura para construção de *No-Fit Polygons*. A segunda contribuição é o desenvolvimento de uma nova heurística de empacotamento de itens irregulares para os problemas *Strip Packing*, *Bin Packing* e *Knapsack*, apresentando, também, uma adaptação entre estes algoritmos.

Para a geração dos *No-Fit Polygons*, são utilizadas duas abordagens geométricas diferentes: a primeira baseada em operações geométricas básicas para a geração dos polígonos; e a segunda baseada na teoria da soma de Minkowski. Implementamos três algoritmos para gerar os *No-Fit Polygons*. O primeiro se baseia no método orbital, enquanto os outros utilizam abordagens distintas da soma de Minkowski.

Em relação aos problemas de empacotamento, propomos uma nova heurística baseada na meta-heurística Algoritmo Genético, gerando a solução por meio de uma abordagem construtiva: dada uma lista de itens a se empacotar, o algoritmo leva em conta a ordem inicial destes e os posiciona no recipiente, considerando algumas de suas características geométricas com a solução parcial. No contexto de busca local do algoritmo, são utilizados operadores para alterar a ordem dos itens na lista, a fim de explorar o espaço de soluções. Como extensão deste trabalho, adaptamos a heurística proposta para os problemas *Bin Packing* e *Knapsack*. Em geral, o algoritmo estendido atua como um compactador dos itens de um recipiente, no momento que a heurística inicial encontra o recipiente cheio.

1.5 Organização

O restante deste trabalho está dividido da seguinte maneira:

O capítulo 2 apresenta as definições básicas utilizadas no trabalho, focando principalmente nas características e definições gerais dos problemas de empacotamento para que possamos descrever os problemas *Strip Packing*, *Knapsack* e *Bin Packing*.

O capítulo 3 apresenta uma breve discussão dos trabalhos relacionados com os resultados mais promissores encontrados na literatura, para os problemas de empacotamento considerados. Além destes, alguns trabalhos que trataram diferentes estruturas geométricas para verificação de sobreposição entre itens irregulares também são apresentados.

O capítulo 4 apresenta algumas técnicas para verificação de sobreposição entre itens irregulares. Dentre elas, destacamos e detalhamos o uso da técnica baseada nos *No-Fit*

Polygons, apresentando três geradores destas estruturas.

O capítulo 5 descreve as heurísticas propostas neste trabalho. É apresentada uma heurística para os problemas de empacotamento com itens irregulares considerados, composta por uma heurística de agrupamento de itens e a pela meta-heurística Algoritmo Genético. Além disso, a proposta de adaptação entre estes algoritmos para a resolução dos problemas *Bin Packing* e *Knapsack* também é descrita.

O capítulo 6 apresenta os resultados computacionais, comparando-se todas as heurísticas implementadas no trabalho. Primeiramente, apresentamos as configurações experimentais adotadas e os resultados para cada um dos geradores de *No-Fit Polygons*. Em seguida, apresentamos essas mesmas informações para cada uma das heurísticas de empacotamento propostas, bem como uma discussão acerca de tais resultados.

Por fim, o capítulo 7 apresenta as conclusões do trabalho, bem como os possíveis trabalhos futuros.

Capítulo 2

Problemas de Empacotamento

Este capítulo apresenta as definições básicas dos problemas de empacotamento, descrevendo as tipologias e variações dos problemas considerados neste trabalho. As definições de alguns conceitos geométricos elementares também são apresentadas.

2.1 Introdução

Por fazer parte do cenário de muitas indústrias, há um estímulo contínuo à novas pesquisas que envolvem problemas de corte e empacotamento, de forma que novas abordagens de geração de solução sejam propostas para tratar adequadamente variações específicas do problema.

Como cada restrição imposta pode resultar em uma variação diferente do problema, a evolução no desenvolvimento das estratégias para obtenção de solução poderá ser prejudicada caso não se adote padrões de classificação dos problemas resolvidos, vista a importância da relação problema/algoritmo.

Com o objetivo de unificar o uso de notações diferentes da literatura e concentrar a investigação sobre diferentes tipos de problemas, Dyckhoff [16] propôs a primeira classificação sistemática para os problemas de corte e empacotamento, descrevendo uma nova tipologia. A grande contribuição de Dyckhoff foi perceber que, dentre os vários problemas de corte e empacotamento, estes apresentavam uma mesma estrutura lógica, de forma que foi possível classificá-los por meio de características comuns. Apesar disso, percebeu-se que a referida tipologia necessitava de alguns ajustes para uma classificação mais geral, sendo feitos recentemente no trabalho de Wäscher *et al.* [43].

Seguindo esta vertente, esse capítulo apresenta os problemas de empacotamento considerandos no trabalho, classificando-os com base na tipologia definida por Wäscher *et al.*.

2.2 Definições básicas

Apesar dos inúmeros tipos e variações, problemas de empacotamento têm como base um conjunto de itens de empacotamento. Como a variação considerada neste trabalho é essencialmente de problemas geométricos e de otimização, utilizamos uma representação baseada no conceito geométrico de polígonos.

Um polígono é uma superfície plana limitada por finitos segmentos de retas (arestas), nos quais cada ponto final destes (vértice) liga-se a, no máximo, dois segmentos (a menos que seja um polígono degenerado, formado por um único ponto). Um vértice de um polígono é dito convexo se o ângulo interno a suas respectivas arestas é menor que 180°. Caso o polígono tenha, no mínimo, um vértice não convexo (chamado também de vértice reflexo), tal polígono é dito não convexo.

O conjunto de arestas de um polígono é chamado de curva poligonal. Pelo Teorema de Jordan, um polígono divide o plano em interior, exterior e borda: interior é a região do plano limitada por curvas poligonais; exterior é a região não limitada por tais curvas; borda é a própria curva poligonal que faz a separação entre interior e exterior.

Um polígono simples é um polígono cujas arestas não adjacentes não se interceptam, ou seja, é uma curva poligonal fechada sem auto-intersecção. Polígonos simples com buracos são topologicamente diferentes, por possuírem curvas poligonais na região interna de uma única curva poligonal externa. No caso dessas curvas poligonais internas, o interior destas representa parte da região externa do polígono. Um exemplo de um polígono simples com buraco pode ser visto na Figura 2.1, sendo este um polígono simples não convexo com dois buracos.

Dado um conjunto de itens em um empacotamento (representados por polígonos), uma das tarefas para resolução do problema é determinar se os itens presentes na solução final não se interceptam uns com os outros. Neste trabalho, definimos que os polígonos se interceptam (sobrepõem-se) caso a região interna de um polígono esteja em contato com com a borda e/ou região interna de outro polígono. Note que o contato entre bordas não implica em intersecção entre polígonos.

As demais definições geométricas utilizadas neste trabalho (envoltória convexa, envoltória retangular, partição convexa de polígonos, soma de vetores, complexidade de algoritmos geométricos etc.) podem ser encontrados em Preparata e Shamos [37].

Por fim, a representação de um item é feita por um polígono simples, composto por uma lista de coordenadas de pontos $((x_1, y_1), (x_2, y_2), \ \ldots \ , (x_n, y_n))$ no plano (represen-

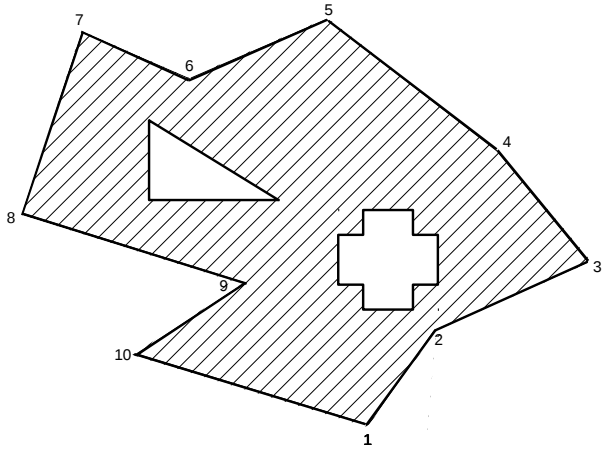


Figura 2.1: Exemplo de polígono simples com buraco, formado por 10 vértices em sua maior curva poligonal.

tando os vértices do polígono), na qual cada um de seus pares consecutivos (inclusive o par $((x_n, y_n), (x_1, y_1))$) representa uma aresta orientada em sentido anti-horário (como apresentado na Figura 2.1). Para as arestas da borda de um polígono, o lado esquerdo desta aresta representa o interior do item. Para as arestas que representam os buracos dos itens (caso existam), o lado esquerdo destas representa o exterior do item.

2.3 Formalização dos problemas de empacotamento

Nos problemas de empacotamento, dados um conjunto R de objetos grandes n-dimensionais de tamanho s também n-dimensional, chamados de *recipientes*, e um conjunto I de objetos menores n-dimensionais, chamados de *itens*, deve-se posicionar os elementos de I em R tal que:

- não haja sobreposição entre quaisquer destes itens; e
- todos os itens de cada recipiente estejam contidos em seu interior,

de forma a maximizar ou minimizar uma dada função objetivo.

Apesar da generalidade da descrição do problema para dimensões arbitrárias, o foco deste trabalho é em problemas de empacotamento 2D. Além disso, nosso interesse é a resolução destes problemas com itens irregulares (a definição para itens irregulares é apresentada na próxima seção), considerando um conjunto limitado de rotações $\theta =$

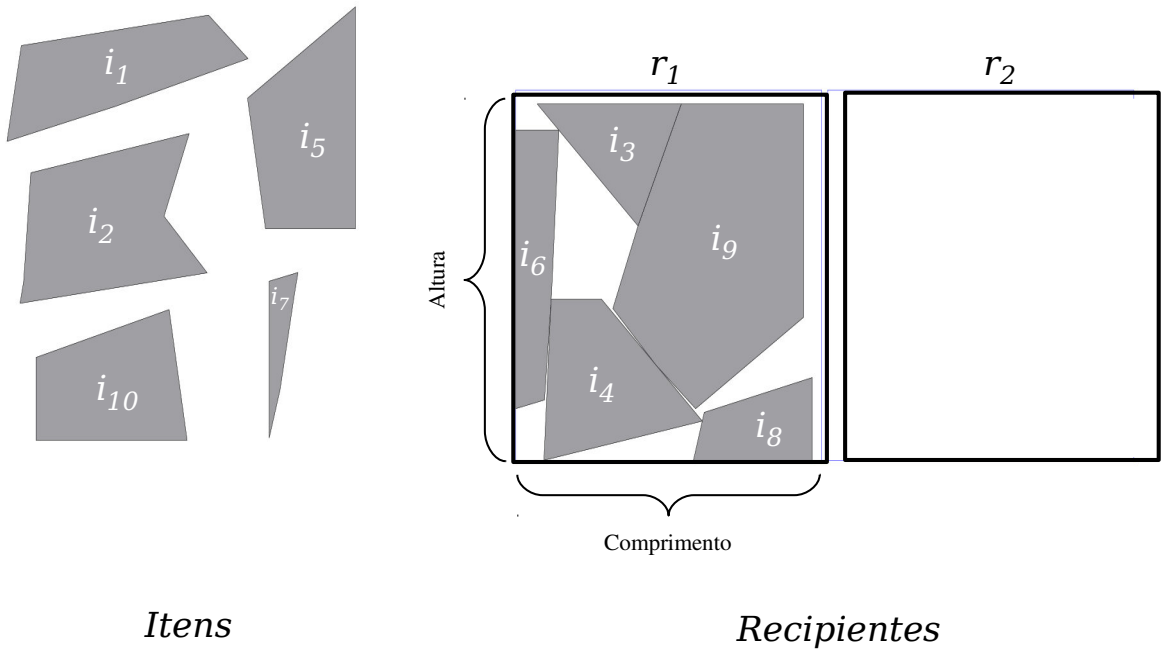


Figura 2.2: Exemplo de empacotamento de itens irregulares em um conjunto arbitrário de recipientes.

$\{\theta_1, \theta_2, \ldots, \theta_k\}, \forall\ k \in \mathbb{N}$, para cada um. Como exemplo, a Figura 2.2 apresenta um empacotamento de um conjunto de itens irregulares distribuídos em um conjunto de recipientes.

2.4 A Tipologia de Wäscher, Haußner e Schumann

A tipologia de Wäscher, Haußner e Schumann [43], utilizada por muitos trabalhos de corte e empacotamento presentes na literatura, foi proposta a fim de cobrir algumas limitações da tipologia descrita por Dyckhoff [16]. Dyckhoff conseguiu descrever a estrutura básica dos problemas de corte e empacotamento, sendo possível agrupar as estratégias de geração de solução desses problemas, até então vistas separadamente. Apesar disso, ao longo dos anos percebeu-se que a sua abrangência era limitada para a classificação de alguns casos, e, atualmente, uma nova tipologia, proposta recentemente por Wäscher *et al.* –que é uma evolução da proposta de Dyckhoff– se mostra mais completa para a classificação dos

problemas de corte e empacotamento.

Dentre as características analisadas para obtenção da tipologia proposta por Wäscher *et al.* estão: formato dos itens utilizados no empacotamento, que podem ser classificados em regulares (todos os itens de um mesmo formato, como retângulos ou círculos ou triângulos etc.) ou irregulares (itens em formato de polígonos simples quaisquer); a dimensão relevante do recipiente e dos itens a serem empacotados, podendo assumir 1, 2, 3, ou mais dimensões; e várias outras características envolvendo os itens, os recipientes etc..

Por si só, a versão unidimensional de muitos problemas de empacotamento pertence a classe de problemas NP-difícil (Garey e Johnson [19]), como, por exemplo, os tratados neste trabalho. No entanto, o tratamento destes fica ainda mais complexo quando trabalhamos com polígonos com buracos ou com itens irregulares.

2.5 Tipos de problemas de empacotamento

Com base na tipologia apresentada por Wäscher *et al.* [43], descrevemos os três problemas básicos utilizados no trabalho, considerando apenas suas versões 2D:

Knapsack. Dados um conjunto I de itens, no qual cada item $i \in I$ apresenta um valor $v(i)$ associado, e o conjunto R formado por um único elemento retangular r (de altura e largura constantes e maiores que zero), busca-se encontrar um subconjunto de I que possa ser empacotado em r de forma que a soma dos valores $v(i)$ dos itens em r seja maximizada, sendo respeitadas as restrições dos problemas de empacotamento. Para o problema considerado neste trabalho, a função $v(i)$ representa a área do item i. Considerando a tipologia de Wäscher *et al.*, este problema está incluído em duas categorias distintas, sendo elas nomeadas "Problema de Posicionamento em Objeto Grande Único" e "Problema *Knapsack* Único". Na primeira categoria é nomeado na literatura como "Problema *Template-Layout*" e, na segunda, como "Problema *Knapsack* Bidimensional" (apesar de não serem considerados itens irregulares nesta).

Problema ***Bin Packing***. Dado um conjunto I de itens, cada item $i \in I$ deve ser empacotado em algum recipiente retangular $r \in R$ (de altura e largura constantes e maiores que zero) de forma a minimizar o número de elementos utilizados de R, sendo respeitadas as restrições dos problemas de empacotamento. Formalmente, isso equivale a geração de uma partição $P = (P_1, P_2, \ \ldots \ , P_k)$ dos itens tal que k seja mínimo e, para cada $P_i, i = 1, \ \ldots \ , k$, todos os itens de P_i podem ser empacotados em um recipiente r. Considerando o conjunto de itens irregulares e recipientes retangulares de dimensão fixa utilizados neste trabalho, é conhecido como "*Bin Packing* Bidimensional Irregular" na literatura, sendo classificado na categoria "Problema *Single Bin-Size Bin Packing*" da tipologia de Wäscher *et al.*.

`Problema` ***`Strip Packing`***. Dado um conjunto I de itens e um único elemento retangular $r \in R$ de altura h, uma constante maior ou igual a zero, e comprimento w, uma variável não negativa, cada item $i \in I$ deve ser empacotado em r de forma a minimizar o valor de w, sendo respeitadas as restrições dos problemas de empacotamento. Nesse caso, dada uma solução para o problema, o valor de w corresponde à diferença da maior coordenada x e a menor coordena x dentre os vértices de todos os itens da referida solução. Este problema está classificado na categoria "Problema de Dimensão Aberta" da tipologia de Wäscher *et al.* e, pelo formato dos itens aqui utilizados, é referenciado na literatura como "Problema *Strip Packing* Irregular" ou "*Nesting Problem*".

Como nosso interesse é na versão irregular destes três problemas apresentados, vamos simplificar a forma de referenciá-los, utilizando, exclusivamente, seus nomes, sem a explicitação de suas possíveis variações, embora estejamos nos referindo às suas versões irregulares. Isso será seguido no restante deste trabalho, a menos que se diga o contrário.

Capítulo 3

Revisão da Literatura

Este capítulo apresenta as principais contribuições encontradas na literatura para resolver os problemas considerados neste trabalho. De modo conjunto, são descritas, também, as técnicas geométricas utilizadas pelas heurísticas propostas para a verificação de sobreposição entre itens do empacotamento.

3.1 Introdução

A área que envolve os problemas de corte e empacotamento vem, ao longo dos anos, despertando interesse nos pesquisadores. Nos últimos anos, houve uma evolução das técnicas desenvolvidas, tanto para a obtenção de soluções destes problemas quanto para um melhor tratamento das estruturas utilizadas durante o processo de otimização.

O tratamento dos problemas de empacotamento utilizando uma abordagem puramente matemática é uma tarefa bastante antiga, porém o desenvolvimento de abordagens automáticas para geração de solução teve início por volta dos anos de 1950, com a publicação dos primeiros trabalhos científicos acerca do assunto (Hopper [25]). Sendo ainda influenciada pelas primeiras estratégias desenvolvidas, tais publicações apresentam um grande teor matemático, com a utilização de modelos lineares para resolução da versão regular destes problemas.

Alguns dos primeiros trabalhos que tratam problemas bidimensionais de posicionamento de itens irregulares foram realizados entre as décadas de 1970 e 1980, quando Adamowicz e Albano [1] descreveram uma técnica de agrupamento de itens irregulares 2D em recipientes retangulares, utilizando um algoritmo baseado na estrutura geométrica *No-Fit Polygon* para a verificação de sobreposição entre os itens. A partir disso, a literatura

começa a apresentar trabalhos que focam nos aspectos geométricos destes problemas, bem como nas técnicas de otimização para se desenvolver novas abordagens para a obtenção de soluções. Uma interessante descrição de tais aspectos para resolução de problemas de posicionamento de itens irregulares pode ser encontrada nos trabalhos de Bennell e Oliveira [3, 4] e de Dowsland e Dowsland [15].

A seguir, destacamos algumas das principais contribuições que encontramos na literatura que tratam problemas de corte e empacotamento com itens irregulares, bem como aquelas para o tratamento geométrico dos itens na construção de um empacotamento.

3.2 Trabalhos relacionados

No trabalho [33], Martins e Tsuzuki apresentam uma nova abordagem de empacotamento de itens irregulares 2D, considerando um conjunto de rotações para os itens e um único recipiente com dimensões fixas, de forma a minimizar as perdas de utilização do recipiente após o empacotamento. A forma de criação da solução é baseada nas estruturas dos *No-Fit Polygons* –seção 4.3–, na qual é calculada a "Região Livre de Colisão" (RLC), onde o próximo item poderá ser empacotado. A RLC é calculada da seguinte forma: dado o próximo item a ser empacotado, calculam-se seus respectivos *No-Fit Polygons*, tanto o resultante com o polígono complemento do recipiente (que representa a região viável dentro do recipiente, se desconsideramos a solução parcial nele contida) –denominado $\overline{\text{NFP}_{RI}}$–, quanto o resultante com polígono que representa a solução parcial do empacotamento (que representa a região inviável, ou seja, a região que gera sobreposição entre itens) –denominado NFP_{PI}–; a RLC corresponde à diferença entre os polígonos gerados, ou seja, $\overline{\text{NFP}_{RI}}$ - NFP_{PI}. Criada uma solução aleatória inicial viável, a evolução da solução é controlada pela meta-heurística *Simulated Annealing*: uma nova solução é gerada por meio de uma heurística que gera soluções vizinhas (soluções geradas pela alteração da posição de um único item de uma solução tomada como base), de forma que a meta-heurística controla a alteração nos parâmetros de empacotamento (ângulo de rotação, parâmetro de posicionamento etc.) do próximo conjunto de itens considerado.

Propondo resolver o mesmo problema, Del Valle *et al.* [14] propõem uma heurística baseada na meta-heurística GRASP, além de uma heurística para a versão irrestrita do problema *Knapsack*. A heurística proposta baseada em GRASP trabalha da seguinte maneira: com base em uma lista inicial de itens, o algoritmo proposto segue uma abordagem construtiva –tipo de abordagem descrita na seção 5.1–, criando uma solução de acordo com a ordem dos itens na lista. A heurística, então, tenta melhorar a solução gerada, realizando uma busca local que altera a ordem dos itens na lista, de forma que novas soluções possam ser formadas. A criação de toda solução é baseada na estrutura dos *No-Fit Polygons*, na qual as posições viáveis para empacotamento dos próximos itens são

determinadas considerando propriedades geométricas do respectivo item com a solução parcial. Baseando-se no mesmo problema resolvido pela heurística GRASP proposta por Del Valle *et al.*, Mundim e Queiroz [34] apresentam uma heurística híbrida que combina a meta-heurística GRASP com a meta-heurística *Simulated Annealing*. A forma de geração da solução proposta por Del Valle *et al.* e por Mundim e Queiroz são muito semelhantes, sendo que este se difere, principalmente, pela aplicação da meta-heurística *Simulated Annealing*, que controla a evolução das soluções da heurística baseada em GRASP, a fim de se realizar uma maior exploração do espaço de soluções.

Terashima-Marín *et al.* [41] apresentam uma abordagem baseada na meta-heurística Algoritmo Genético para criação de hiper-heurísticas para o problema *Bin Packing* irregular, considerando instâncias com itens regulares (itens com formato retangular) e irregulares (itens em formato de polígono convexo). O objetivo das hiper-heurísticas é decidir quando e onde aplicar heurísticas básicas sobre um conjunto de itens de empacotamento. Nesse caso, as heurísticas básicas se referem à heurísticas para seleção de itens e recipientes para empacotamento, além de heurísticas de posicionamento de um item sobre a solução parcial. A criação das hiper-heurísticas do trabalho (feita pelo método baseado em Algoritmo Genético) usa um modelo de solução baseado em uma representação de comprimento variável, que corresponde à combinações de regras do tipo condição/ação, utilizadas pela hiper-heurística gerada ao construir uma solução. Seguindo a mesma linha de resolução com o uso de hiper-heurísticas para resolução do problema *Bin Packing*, López-Camacho *et al.* [31] propõem uma nova estrutura de representação deste modelo de otimização, a fim de se melhorar a capacidade de geração de boas soluções por meio destas heurísticas.

As mais variadas estratégias encontradas na literatura para a resolução de problemas de empacotamento foram as propostas para o problema *Strip Packing*. Dentre elas, a maior parte se baseia no uso de meta-heurísticas como forma de guiar as heurísticas de empacotamento de itens na construção de soluções, com destaque para os trabalhos [7, 8, 9, 22, 6, 27, 42, 29, 39].

No trabalho [7], Burke *et al.* apresentam uma nova heurística para o problema de Corte de Estoque bidimensional com itens irregulares. Primeiramente, foi apresentada uma nova técnica de resolução de sobreposição entre os itens. O principal motivo disso foi a necessidade de se utilizar técnicas mais precisas para resolver os problemas de sobreposição entre itens irregulares, além de algumas limitações apresentadas pelos métodos da literatura, que até então não trabalhavam com itens com arcos e/ou buracos de modo eficiente. Assim, a proposta foi trabalhar com as primitivas dos itens, formadas por suas arestas e/ou arcos. Para resolver a sobreposição entre duas primitivas de itens diferentes, aplica-se um deslocamento vertical em uma das primitivas na direção positiva do eixo y (altura) do recipiente, sendo este deslocamento representado pela distância necessária

para se remover a sobreposição. Se uma primitiva de um item é deslocada, as demais primitivas do respectivo item sofrem o mesmo deslocamento. Este processo é repetido até que todas as primitivas de dois itens diferentes não mais se sobreponham. Devido o deslocamento sobre o eixo y não ser feito de maneira discreta, mas de forma contínua –por considerar as próprias primitivas dos itens–, a precisão do processo de resolução de sobreposição foi, de fato, melhorada. No entanto, o eixo x ainda é considerado de forma discreta, caso nenhuma região viável dentro do recipiente seja encontrada ao longo do eixo y, considerando um determinado valor de x para um item. Nesse caso, como o objetivo é posicionar o próximo item de forma que se utilize o menor comprimento do recipiente, são feitos incrementos em x para a análise do eixo y até que se resolva a sobreposição, de modo que a qualidade do empacotamento e velocidade de geração de solução estão diretamente ligados ao tamanho dos intervalos adotados nesta discretização. Os autores, então, apresentam uma heurística para o problema, baseada na estratégia *Bottom-Left*, utilizando a técnica de resolução de sobreposição proposta. Juntamente com a heurística desenvolvida, Burke *et al.* também aplicaram dois métodos de busca para alterar a sequência de itens passada como entrada para o algoritmo: o *Hill Climbing*, que aplica operadores de troca de posição dos itens da solução atual a fim de encontrar uma solução vizinha de melhor qualidade; e a Busca Tabu, que trabalha gerando um dado número de soluções vizinhas que não estejam presentes em uma lista (chamada de lista tabu), o que permite uma busca de soluções mais diversificada.

Já no trabalho [8], Burke *et al.* apresentam uma forma mais robusta para lidar com a resolução de sobreposição entre itens, em substituição à técnica anteriormente apresentada (Burke *et al.* [7]). Basicamente, uma nova forma de criação das estruturas dos *No-Fit Polygons* é descrita, tratando todos os casos degenerados entre os itens, como, por exemplo, os casos de itens com buracos e concavidades. Todo este processo está estritamente ligado a itens compostos por segmentos de retas, sendo que a geometria de itens compostos por curvas deve ser aproximada por múltiplos destes segmentos. Os autores apresentam resultados para a criação de *No-Fit Polygons*, destacando a eficiência em se utilizar esta técnica. Como uma extensão deste trabalho, Burke *et al.* [9] generalizam sua técnica de criação de *No-Fit Polygons* para itens representados por arcos e curvas, considerando, também, todos os casos possíveis. Ao final, os autores realizam um conjunto de testes, utilizando a mesma heurística de empacotamento e critérios de busca adotados no trabalho [7], com a adição das técnicas de criação e verificação de sobreposição com *No-Fit Polygons*.

Outro algoritmo desenvolvido que pode ser utilizado para tratar instâncias do problema *Strip Packing* com itens irregulares foi proposto por Gomes e Oliveira [22], desconsiderando buracos nos itens. Neste trabalho, similar ao trabalho de Bennell e Dowsland [2], foi apresentado um método de resolução do problema baseado em programação li-

near, porém, hibridizando-o com a meta-heurística *Simulated Annealing*. O modelo de programação linear é utilizado em dois métodos distintos. O primeiro, responsável por melhorar uma solução viável, é o algoritmo de *compactação*. Neste método, se a solução pode ser melhorada, o programa linear encontrará tal solução, sendo esta uma solução ótima local. O outro é o método de *separação*, que tenta encontrar uma solução viável, dada uma solução inviável (com sobreposições) –tipo de abordagem descrita na seção 5.1, descrita como "Redução de Sobreposição"–. Em relação a busca de soluções, o algoritmo baseado na meta-heurística *Simulated Annealing* guia um processo de construção de soluções vizinhas. Os autores apresentam a estrutura de vizinhança utilizada pelo método proposto, que, resumidamente, é capaz de fazer a troca de dois itens em uma solução, considerando todas as rotações permitidas entre estes. A última consideração feita por Gomes e Oliveira refere-se a uma estratégia multi-estágio para algumas instâncias: os itens foram divididos em grupos (considerando a área destes) para que fossem empacotados em estágios diferentes. Nesse trabalho, o *No-Fit Polygon* é a técnica básica para verificação de sobreposições entre os itens.

A proposta elaborada por Egeblad *et al.* [17] utiliza uma técnica geométrica específica para verificação de sobreposição e uma meta-heurística de busca para resolver o problema *Strip Packing* bidimensional para itens irregulares com um conjunto finito de rotações. A heurística proposta é baseada em uma busca local que utiliza a abordagem de redução de sobreposição para a criação de uma solução vizinha. A geração de soluções vizinhas é feita da seguinte forma: tomando um item P que sobrepõe outro(s) item(ns), encontra-se a movimentação (transladação, rotação ou inversão) de P que minimiza a sua quantidade de sobreposição com os demais itens, movendo-o para esta posição. Este processo tem uma complexidade de tempo $O(mn \log(mn))$, sendo m o número de arestas de P e n o número de arestas dos demais itens. A forma de verificação de intersecção entre dois itens é baseada no "Teorema de Intersecção de Área", descrito por Nielsen e Odgaard [35]. Outro ponto fundamental da heurística proposta é o modo de fuga de mínimos locais, utilizando um algoritmo baseado na meta-heurística "Busca Local Guiada", que basicamente trabalha levando em consideração as sobreposições dos itens de um empacotamento. Por fim, os autores estendem seu trabalho para o caso tridimensional do problema.

Algumas propostas que se baseiam em programação não linear para resolver o problema *Strip Packing* bidimensional com itens irregulares são apresentadas nos trabalhos [27], [42] e [29]. Tais propostas foram baseadas em uma busca local que faz a movimentação e a troca de itens dentro do recipiente, de forma a se encontrar posições com a menor quantidade de sobreposição entre eles, aplicando-se, em seguida, o método não linear de separação. No entanto, acabam diferindo-se em alguns pontos específicos.

No trabalho de Imamichi *et al.* [27], o algoritmo proposto para o problema é dividido em dois métodos básicos. O primeiro, no qual a programação não linear é aplicada, é

o método de *separação* –descrito anteriormente–. Neste método, a rotação dos itens é fixada, assim como o comprimento do recipiente. Se uma solução encontrada não for viável, incrementa-se o tamanho do recipiente até que se tenha uma solução viável. Após esta busca, é retornada uma solução ótima local para o empacotamento. O segundo método faz uma perturbação em uma solução atual, de modo que dois itens sejam trocados de posição dentro de uma solução. Durante este processo de troca, todas as rotações permitidas do item considerado são testadas (as rotações dos demais itens são mantidas fixas) e a melhor posição encontrada, ou seja, aquela que gera um posicionamento em que não haja sobreposição ou que esta seja a menor (caso não haja posições viáveis de empacotamento), é escolhida. Toda verificação de sobreposição entre itens é baseada na técnica de *No-Fit Polygons*.

Já a proposta de Umetani *et al.* [42] aplica uma busca diferente, na qual há a transladação de um polígono que sobrepõe outro por meio de movimentos sucessivos na horizontal e na vertical sobre uma solução, até que se remova (ou se reduza) as possíveis sobreposições. Nesse caso, os autores avaliam a quantidade de sobreposição utilizando uma técnica chamada *Penetration Depth* direcional, que indica a menor distância horizontal ou vertical para se remover a sobreposição entre dois itens dentro do recipiente, utilizando, para isso, o *No-Fit Polygon* dos itens considerados. Por fim, como forma de guiar a geração de novas soluções do empacotamento, a heurística proposta é baseada na meta-heurística Busca Local Guiada, que visa movimentar itens da solução que têm uma maior probabilidade de gerar melhores soluções.

Alguns dos melhores resultados de utilização de recipiente para o problema *Strip Packing* são apresentados no trabalho de Leung *et al.* [29], no qual propõem um modelo não linear de otimização com uma busca local baseada na meta-heurística Busca Tabu. A proposta de Leung *et al.* teve como base o trabalho de Imamichi *et al.* [27]. A diferença acerca destes dois trabalhos está relacionado à meta-heurística utilizada que guia a busca de soluções vizinhas. A heurística implementada, que é baseada na Busca Tabu, toma a lista de itens do empacotamento e controla a movimentação destes com a lista tabu, estabelecendo que os itens presentes nesta lista permaneçam períodos de tempo sem movimentação, de modo que toda a solução seja explorada. Ao final, Leung *et al.* ainda aplicam um algoritmo de compactação da solução, a fim de evitar que hajam espaços livres no recipiente, criados pelo método de separação. Ressaltamos que toda busca local realizada tem como base, também, a utilização dos *No-Fit Polygons*.

Sato *et al.* [39] propõem uma nova heurística construtiva, que é direcionada por uma heurística baseada na meta-heurística *Simulated Annealing* para resolver o problema *Strip Packing* irregular. Em sua proposta, os autores descrevem um algoritmo construtivo que se baseia em uma lista de itens de empacotamento. A heurística de agrupamento dos itens tem como ponto principal a RLC –Região Livre de Colisão, descrita anteriormente–,

de forma que é considerado um recipiente de dimensões fixas. O algoritmo de agrupamento escolhe a posição de empacotamento do próximo item selecionando uma posição de interesse sobre a borda da RLC, com a seguinte prioridade de escolha: vértice degenerado (tratado na seção 4.3 como ponto interior); aresta degenerada (tratada na seção 4.3 como aresta coincidente); e vértice convexo. Devida esta prioridade, os autores destacam a importância da geração de tais estruturas degeneradas durante o empacotamento. A heurística final é completada com o modelo de busca descrito pela meta-heurística *Simulated Annealing*, que trabalha em dois níveis de execução: o primeiro realiza o empacotamento; e o segundo controla a construção das soluções mediante o comprimento do recipiente, podendo incrementar ou decrementar este valor.

Por fim, o trabalho de Bennell e Song [6] propõe uma nova abordagem construtiva de soluções, utilizando como base a heurística TOPOS de Oliveira *et al.* [36]. No entanto, tal heurística sofreu uma revisão em vários pontos, entre eles a utilização de um novo gerador de *No-Fit Polygons* para o processo, novas formas de avaliação da solução parcial etc.. Considerando a heurística de agrupamento, a escolha do próximo item pode ser feita de duas formas: por busca local, no qual o próximo item agrupado será aquele que, dentre todos itens ainda não empacotados, gerar o melhor benefício, de acordo com o critério de avaliação estabelecido; ou por uma ordenação inicial preestabelecida, que não realiza avaliação entre itens, mas baseia-se apenas na ordenação inicial dos mesmos. Dada esta heurística e a lista de itens a ser empacotada, a heurística completa toma cada item desta lista e os empacota no recipiente, utilizando uma heurística de agrupamento. A ordem dos itens dessa lista é uma característica importante do método, e está vinculada à heurística *Beam Search*, que usa uma estrutura de árvore (com nós e ramos), fazendo uma busca semelhante ao método de busca *branch-and-bound*, e é responsável por controlar a busca das soluções, realizando as podas de ramos de acordo com a função de avaliação, ao considerar determinado nó.

3.3 Resumo das técnicas encontradas

Como visto, apesar da distinção de cada uma das heurísticas para resolução dos problemas de empacotamento considerados neste trabalho, é importante destacar que todas as propostas citadas buscam utilizar mecanismos eficientes de verificação de sobreposição entre os itens de um empacotamento para a construção de suas soluções. Note que, por utilizarem um conjunto fixo de rotações para os itens, a maior parte dos trabalhos convergem à utilização da técnica baseada nos *No-Fit Polygons*, utilizando-os de forma direta ou indireta à cada item considerado no processo de empacotamento.

Outro ponto importante se relaciona à constituição das heurísticas propostas. Como o problema de empacotamento é NP-difícil, as heurísticas propostas, na maior parte dos

trabalhos encontrados na literatura, estão ligadas à meta-heurísticas. A grande flexibilidade proporcionada para construção das soluções é uma característica importante destas técnicas de busca, vista a dificuldade de se determinar regras fixas para o empacotamento que resultem sempre em bons resultados.

Capítulo 4

Foco geométrico: a verificação de sobreposição e os *No-Fit Polygons*

Este capítulo apresenta algumas das formas de se tratar sobreposição entre itens irregulares. Destacamos algumas opções presentes na literatura, focando no modelo utilizado neste trabalho: a verificação baseada nas estruturas dos No-Fit Polygons*. São apresentados, também, algoritmos da literatura para geração destas estruturas, baseados em operações geométricas básicas e na soma de Minkowski.*

4.1 Introdução

Quando estamos trabalhando com problemas de empacotamento, uma das questões fundamentais é a forma que itens e recipientes podem assumir neste processo. Notoriamente, esta é apenas uma de várias restrições que, quando imposta ao problema original, acaba elevando o seu grau de dificuldade de resolução.

Diante disso, quando consideramos que os itens utilizados em um empacotamento possuem um formato com característica irregular –seção 2.4–, determinar a disposição de dois destes (i.e., se se sobrepõem, se estão apenas se tocando ou se estão separados) é uma tarefa complexa, devido a estrutura geométrica envolvida. Aqui, diferentemente do caso de itens regulares, a complexidade das operações de verificação de sobreposição entre itens não é constante, mas está diretamente ligada a estrutura geométrica dos polígonos que os representa.

A literatura apresenta uma grande variedade de técnicas para verificação de sobreposição entre itens de um empacotamento, quando esses assumem formato irregular. É importante notar que, geralmente, cada técnica adota uma abordagem geométrica diferente, de maneira que possa ou melhorar a eficiência da verificação ou simplificar a forma de tratamento dos itens, dependendo da abordagem seguida.

4.2 Verificação de sobreposição entre itens irregulares

Considerando itens em seu formato irregular, a verificação de sobreposição entre estes pode ser feita de diversas formas. Destacamos algumas apresentadas por Bennell e Oliveira [3].

A primeira, denominada de método *Raster*, faz a discretização das posições do recipiente, ou seja, tem-se uma aproximação de um cenário contínuo para um cenário discreto. Da mesma forma, itens de um empacotamento passam a adotar uma representação similar a esta. Representando itens e recipientes com o uso de uma matriz de pontos, tem-se uma redução significativa da quantidade de informação para empacotamento, fazendo com que, consequentemente, as implementações baseadas neste modelo sejam bastante eficientes. No entanto, o nível de discretização aplicada acaba ocasionando o detrimento da qualidade solução, pelo fato de se limitar o conjunto de posições viáveis durante o empacotamento.

Outra maneira de se verificar sobreposições é com o uso direto de operações trigonométricas. Nesse caso, toda verificação é feita com base em testes de intersecção de arestas e localização de ponto no plano. Embora este método seja mais preciso que o método *Raster* e existam algumas verificações sobre os pares de itens (e.g., com o uso das "funções-D" –Konopasek [28]–) que evitam a utilização de todas as arestas de cada item da solução, o tempo gasto para verificação das sobreposições é muito maior em relação ao número de arestas. Logo, torna-se inviável aplicá-la quando o intuito é gerar a solução de forma rápida, levando em conta a versão irregular dos problemas de empacotamento com muitos itens.

Um terceiro método é o que utiliza o chamado *No-Fit Polygon*. O uso do *No-Fit Polygon*, resumidamente, faz com que a verificação de sobreposição entre os dois itens se reduza a um problema de verificação de ponto em um polígono simples. O uso destas estruturas apresenta uma série de vantagens quando utilizamos um conjunto limitado de rotações para os itens. Por exemplo, com o pré-processamento de cada par de *No-Fit Polygon*, o processo de verificação de sobreposição durante um empacotamento terá uma redução no tempo gasto, se comparado com a abordagem trigonométrica, que deve, repetidamente, detectar e resolver a mesma verificação de sobreposição dos itens em repetidas orientações

e posições no recipiente. Como, neste trabalho, a forma de verificação de sobreposição entre itens é baseada no uso dos *No-Fit Polygons*, detalhamos suas características na próxima seção.

4.3 A estrutura geométrica dos *No-Fit Polygons*

O *No-Fit Polygon* (NFP) é uma estrutura geométrica que permite definir posições em que dois polígonos simples podem ser posicionados sem que haja sobreposição entre eles. Assim, considerando-os em problemas de empacotamento, são muito utilizados para a verificação de sobreposições entre itens e para a descrição de pontos viáveis de empacotamento dentro do recipiente. De forma geral, é uma boa opção para ser utilizada em problemas de empacotamento com itens irregulares por ser base para técnicas eficientes de verificação de sobreposição entre itens (em comparação com as operações trigonométricas elementares para tal verificação), além de ser precisa (por trabalhar com as próprias arestas dos polígonos) e ainda permitir o desenvolvimento de heurísticas que possam tirar proveito dessas estruturas durante um empacotamento (Bennell e Oliveira [3]).

Gerado o NFP do item A para o item B, representado por NFP_{AB}, a verificação de sobreposição entre os dois itens se reduz a uma problema de determinar se o ponto de referência de B está contido no interior do NFP_{AB} (uma operação de complexidade $O(n)$, no qual n é o número de arestas do NFP_{AB}, ou seja, a complexidade de se determinar a localização de um ponto em um polígono simples). Determinada a localização do ponto, é trivial decidir se há ou não sobreposição entre os itens: se o ponto está contido no interior do respectivo NFP, há sobreposição. Caso contrário, o posicionamento entre estes não gera sobreposição, podendo estarem disjuntos (ponto de referência de B fora do NFP_{AB}) ou em contato (ponto de referência de B sobre uma das arestas de NFP_{AB}). Este processo que envolve operações sobre NFPs pode ser visto na Figura 4.1.

A escolha de um algoritmo para gerar os NFPs é uma tarefa importante, pois a complexidade de tempo de cada um pode ser diferente, de acordo com a sua forma de tratar as operações geométricas envolvidas nesse processo. Utilizar um gerador suficientemente robusto, de forma que seja rápido na execução e que trate todos os tipos de polígonos, sejam estes não convexos e/ou com buracos, é algo fundamental, embora essa seja, talvez, a meta mais difícil de se alcançar.

Diante disso, apresentamos três algoritmos que constroem NFPs, com base em duas abordagens geométricas distintas. Neste caso, o objetivo é realizar uma análise empírica das técnicas de geração implementadas.

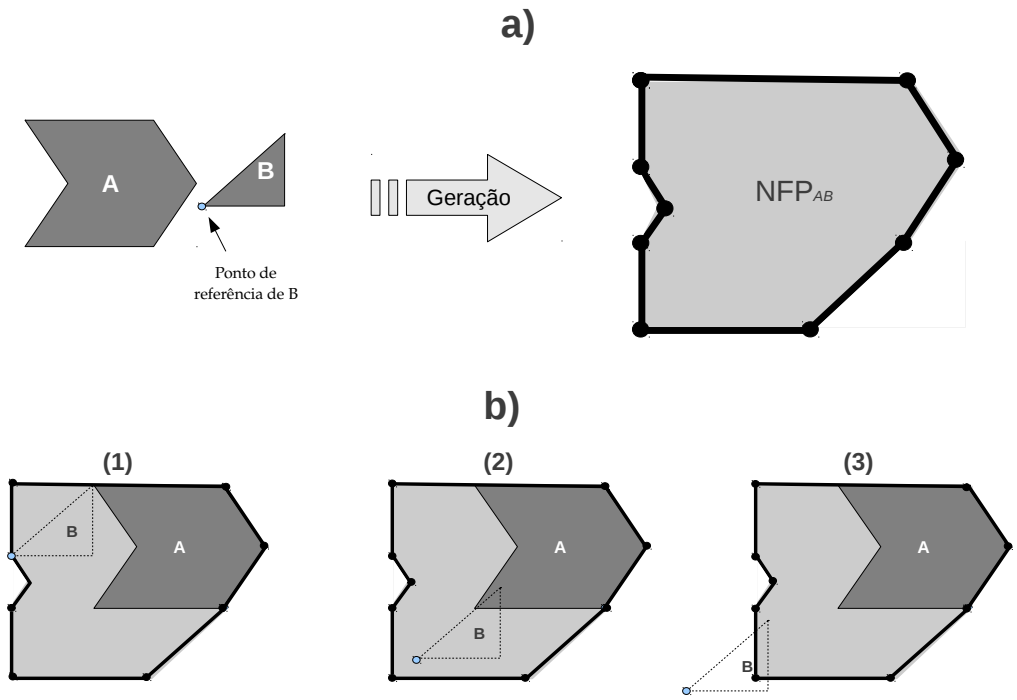


Figura 4.1: NFP: a) Processo de concepção; b) Processo de verificação, com (1) polígonos em contato pela borda, (2) polígonos sobrepostos e (3) polígonos separados.

4.3.1 Abordagens geométricas de geração de NFPs

O método orbital

Proposto por Mahadevan [32], o modelo orbital utiliza a movimentação de polígonos e trigonometria elementar da seguinte forma: dados dois polígonos simples A e B, um ponto (geralmente um vértice) de cada um é escolhido, sendo chamados pontos de referência; o polígono A é transladado no espaço de coordenadas de forma a posicionar o seu ponto de referência na origem, e este polígono se mantém fixo durante todo o processo; já o polígono B, deve ser deslocado em torno de todas as arestas de A, mantendo sempre o contato entre arestas de ambos os polígonos (sem que haja sobreposição), e, à medida que isso é feito, traçar cada aresta da borda do NFP, baseando-se no caminho feito pelo ponto de referência de B. Este processo pode ser visto na Figura 4.2, em um processo de construção que vai de I até IX, sendo X a representação do NFP$_{AB}$.

O algoritmo implementado para esta abordagem de criação de NFPs é o proposto por Burke *et al.* [8]. A característica a se destacar deste algoritmo é a sua capacidade de gerar o NFP para qualquer tipo de polígono simples, com buracos ou não. Outro ponto importante se refere à capacidade de encontrar bordas do NFP formadas por *arestas*

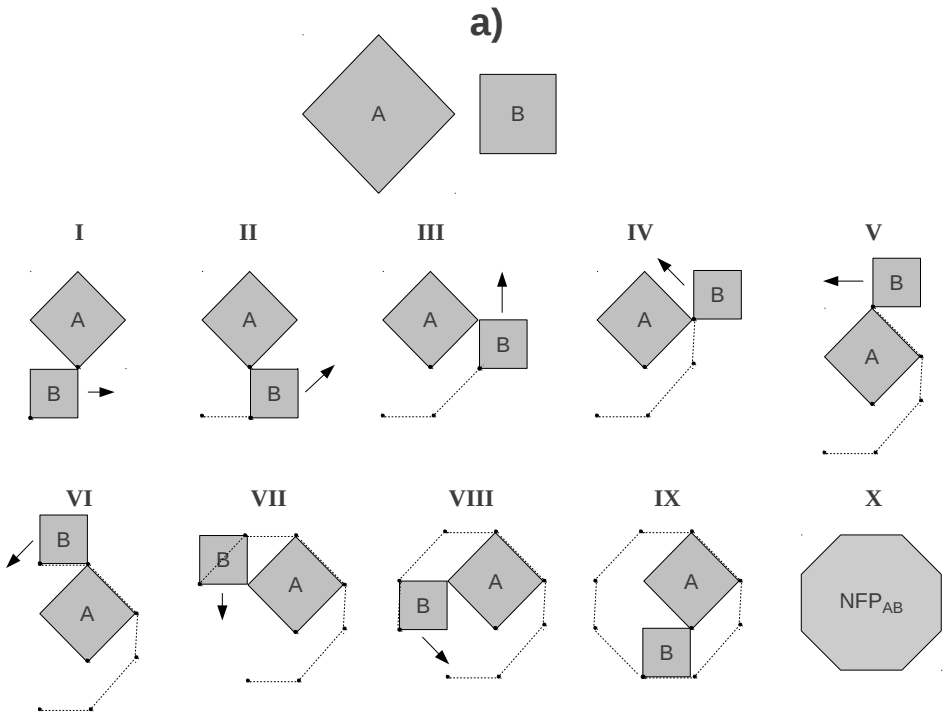


Figura 4.2: Processo de construção de NFP pelo método orbital: a) Polígonos A e B utilizados; I-IX: passos de construção; X: NFP$_{AB}$ resultante.

coincidentes (ou porções destas) ou simples *pontos interiores*, que são regiões nas quais, uma vez posicionados corretamente, os polígonos base do NFP se encaixam exatamente, mantendo contato por no mínimo duas arestas. Um exemplo de NFP que possui estas características é apresentado pela Figura 4.3. A geração destas arestas do NFP é importante para qualquer algoritmo de empacotamento, visto que a escolha da posição de um item no recipiente considerando estas regiões tende a gerar soluções mais compactas.

A soma de Minkowski

A soma de Minkowski (SM), fundamentada na teoria dos conjuntos e na geometria integral, é uma das operações elementares da área de processamento de imagens, constituindo parte da chamada morfologia matemática. Mais especificamente, corresponde à operação de dilatação.

Sejam A e B dois conjuntos de pontos no espaço $\mathbb{R}^d$. A SM de A com B é definida como (4.1):

$$A \oplus B = \{a + b \mid a \in A, b \in B\}, \tag{4.1}$$

nos quais a e b são vetores correspondentes a cada ponto de A e de B, respectivamente,

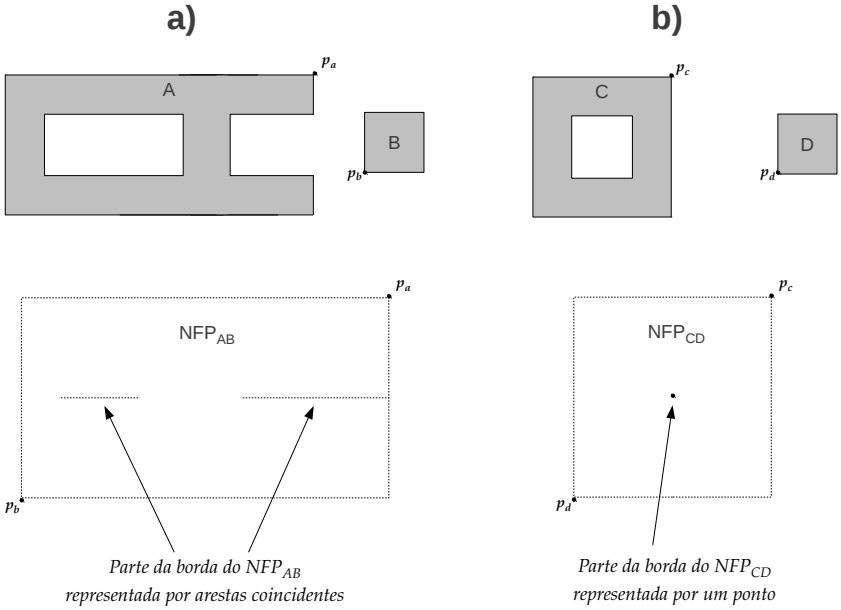


Figura 4.3: NFP composto por: a) arestas coincidentes; e b) ponto interior.

e $a + b$ é a soma vetorial de a e b.

Dado que o interesse é utilizar a teoria da SM no processo de verificação de sobreposição entre polígonos simples, a forma de se aplicar seus conceitos sobre dois conjuntos de pontos A e B para a criação do NFP_{AB} é descrita no Teorema 1:

Teorema 1 (Detecção de intersecção de conjuntos pela soma de Minkowski)**.** *Sejam A e B dois conjuntos de pontos em $\mathbb{R}^2$ e $p = (p_x, p_y)$ um ponto no plano, representando um vetor de deslocamento que sai da origem e termina em (p_x, p_y). Seja B_p o conjunto B transladado pelo vetor p, então $A \cap B_p \neq \emptyset$ se, e somente se, $p \in A \oplus (-B)$, no qual $(-B) = \{-b \mid b \in B\}$ é o conjunto simétrico de B em relação à origem.*

Demonstração. ($\Rightarrow$) Suponha que $x \in A \cap B_p$. Com isso, $x \in A$ e $x \in B_p$, o que significa que existe $b \in B$ tal que $x = b + p$. Assim, temos $p = x + (-b)$, que representa um ponto em $A \oplus (-B)$. Logo, $p \in A \oplus (-B)$.

($\Leftarrow$) Suponha que $p \in A \oplus (-B)$. Logo, temos $p = a + (-b)$ para $a \in A$ e $b \in B$. Reformulando esta expressão como $a = p + b$, a pertence tanto a A quanto a B_p, o que implica $A \cap B_p \neq \emptyset$. □

A interpretação para a geração do NFP a partir da SM é a seguinte: a SM dos pontos de A com o conjunto simétrico $(-B)$ representa o conjunto no qual cada vetor destes pontos (vetor de ponto inicial na origem) poderá deslocar B de forma que este conjunto sobreponha A. Note que, independentemente da posição de A e B no plano, os conjuntos

gerados pela operação $A \oplus (-B)$ serão congruentes[1] (nesse caso, o conceito geométrico de congruência se aplica aos conjuntos gerados utilizando-se apenas a translação), sendo que A e B_p irão se sobrepor quando o resultado da SM contiver, obrigatoriamente, o ponto de origem do plano. Isso vem do fato de que se A e B_p se sobrepõem, então existe $x \in A \cap B_p$ e, ao computar a SM de A e $-B_p$, somaremos x e $-x$.

Quando consideramos que os conjuntos A e B são polígonos ou objetos não poligonais (e.g, círculos, elipses etc.), deve-se adaptar a forma de aplicação da SM para a geração do NFP_{AB}, pelo fato de tais objetos serem formados por infinitos pontos. Diante disso, Cunninghame-Green [13] e Ghosh [20, 21] apresentaram uma forma de se calcular a SM, levando-se em consideração apenas as arestas dos polígonos. De forma semelhante, é necessário o uso do conjunto simétrico $(-B)$ para se obter o respectivo NFP. Assim, como o caso em que consideramos um conjunto de pontos, o simétrico do polígono B pode ser representado por $-B$, e, nesse caso, corresponde à inversão do sentido de cada uma de suas arestas (troca entre os pontos inicial e final da aresta), mas sem a alteração de sua orientação topológica inicial (ordem de conexão entre as arestas). De maneira representativa, para o polígono

$$B = (\overrightarrow{b_1} \rightarrow \overrightarrow{b_2} \rightarrow \cdots \rightarrow \overrightarrow{b_n}),$$

temos a representação do seu simétrico como

$$-B = (\overleftarrow{b_1} \rightarrow \overleftarrow{b_2} \rightarrow \cdots \rightarrow \overleftarrow{b_n}),$$

nos quais "$\rightarrow$" representa a sequência da ordem topológica das arestas, e "$\overrightarrow{b_x}$" e "$\overleftarrow{b_x}$" representam o sentido (normal ou inverso) da aresta x de B, considerando o sentido das arestas de B.

Com base na teoria da SM, implementamos duas técnicas para gerar NFPs de polígonos:

Decomposição

Trabalha com o conjunto de vértices de polígonos simples. Para gerar NFP com esta técnica, a ideia é decompor polígonos simples não convexos em polígonos simples convexos, computar a SM entre os pares de polígonos da decomposição e uní-las: sejam A e B dois polígonos simples, obtemos os polígonos convexos $A_1, \ldots, A_k$ e $B_1, \ldots, B_\ell$ tal que $\bigcup_{i=1}^{k} A_i = A$ e $\bigcup_{j=1}^{\ell} B_j = B$. De posse desses conjuntos, calculamos a SM $S_{ij} = A_i \oplus -B_j$ para cada par, obtendo m polígonos convexos. Finalmente, o NFP_{AB} é obtido pela união

[1]Dois conjuntos de pontos geométricos são ditos congruentes se, e somente se, um pode ser transformado no outro pela combinação de translações e/ou rotações e/ou reflexões.

destes m polígonos, ou seja, $\text{NFP}_{AB} = A \oplus -B = \bigcup_{ij} S_{ij}$. O que podemos notar é que a complexidade de tempo deste método pode depender da complexidade do algoritmo usado para decomposição convexa de polígonos simples.

Seguindo a técnica de decomposição para o cálculo de NFPs, implementamos um algoritmo baseado no método de SM da biblioteca geométrica CGAL [10], vista na seção 6.2.2, fazendo uso da decomposição convexa proposta por Chazelle e Dobkin [11], de complexidade $O(n^2)$, no qual n é o número de vértices do polígono. A característica do algoritmo implementado é a geração de NFPs apenas para instâncias formadas por polígonos simples sem buracos, sendo esta uma restrição imposta pela biblioteca CGAL em seu algoritmo de cálculo da SM entre polígonos. Outra restrição se refere à incapacidade de encontrar arestas coincidentes e pontos interiores no NFP, que são eliminados durante as operações de união dos polígonos.

Adição de Borda

A Adição de Borda (AB) trabalha com as arestas dos polígonos utilizados. Apesar de ser formalizada por Ghosh [20], o trabalho de Cunninghame-Green [13] já apresentava a utilização de ideias da AB para a geração do NFP de polígonos convexos para produzir o chamado "espaço de configuração dos obstáculos". Na próxima seção, fazemos uma discussão sobre a teoria da Adição de Borda para geração de NFPs.

4.3.2 A Adição de Borda

Trabalhar com polígonos para resolver um problema geométrico é interessante em muitas ocasiões, visto sua simplicidade e facilidade de representação computacional. Apesar disso, mesmo trabalhando apenas com as bordas dos polígonos, a aplicação dos conceitos de AB para o cálculo da SM não é uma tarefa trivial, em grande parte dos casos. Entretanto, se os polígonos apresentam alguma característica geométrica (como convexidade, por exemplo), este cálculo pode ser facilitado.

Visando estender o conceito da SM para os elementos constituintes de um polígono (vértices e arestas), Ghosh [20] apresentou uma série de operações sobre os objetos poligonais, agrupando-os na chamada *Teoria de Adição de Borda*, que podem ser aplicadas aos polígonos simples sem buraco. A seguir, fazemos uma apresentação das técnicas baseadas nesta teoria, modelando-as a fim de se obter o respectivo NFP dos polígonos envolvidos.

A Adição de Borda considerando polígonos convexos

Ghosh [20] propõe, inicialmente, a teoria da AB para o caso de polígonos convexos, baseando-se nos resultados de Grünbaum *et al.* [23] para a realização da SM para politopos convexos d-dimensionais, em que d é uma dimensão arbitrária do espaço euclidiano. Diante da proposta de Ghosh, considerando os polígonos convexos A e B, a ideia principal pode ser resumida nos seguintes fatos:

- *Soma de Minkowski de dois pontos*: somar vetorialmente dois pontos (vértices) resulta em um ponto (vértice), isto é, $v\acute{e}rtice_A + v\acute{e}rtice_B$;
- *Soma de Minkowski de um ponto e um segmento de reta*: somar um ponto (vértice) e uma aresta resulta na mesma aresta transladada pelo vetor deste ponto (vértice), isto é, $v\acute{e}rtice_A \oplus aresta_B$ ou $aresta_A \oplus v\acute{e}rtice_B$;
- *Soma de Minkowski de dois segmentos de retas*: somar duas arestas paralelas resulta em uma aresta paralela às anteriores, cujo comprimento corresponde à soma do comprimento destas arestas, isto é, $aresta_A \oplus aresta_B$.

Este processo pode sofrer ainda mais simplificação, se o que interessa é a forma resultante desta soma, e não a sua posição no sistema de coordenadas. Logo, considerando que $S = A \oplus -B$, apenas duas informações são necessárias: (i) quais arestas estão em S; e (ii) a informação topológica das arestas. Diante disso, a ordem de escolha de quais arestas irão compor S se baseia em dois fatos:

1º Seja A um polígono convexo, representado pelo conjunto de suas arestas consecutivas $\{a_1, a_2, \ldots, a_n\}$. Devido à convexidade, estas arestas ocorrem em uma *ordem angular ordenada*; o mesmo acontece para $-B$; e

2º Se A e $-B$ são ambos convexos, sua SM também será convexa.

Uma explicação detalhada para estes fatos pode ser encontrada em Grünbaum *et al.* [23]. No entanto, podemos inferir uma característica diante de tais fatos apresentados: se o resultado obtido para a AB deve ser igual à SM de um conjunto de pontos, as arestas de S também seguirá uma ordem angular ordenada, que só poderá ser obtida com a ordenação e concatenação das arestas de A e de $-B$. Com isso, os passos necessários para o cálculo da AB de dois polígonos convexos A e $-B$, que corresponde à SM destes, podem ser descritos em um procedimento, que está exposto no Algoritmo 1. Em resumo, a teoria da AB para polígonos convexos reduz o problema de se computar $S = A \oplus -B$ a duas operações básicas com números reais: *ordenação* e *soma*.

Algoritmo 1: AdicaoDeBorda_PoligonosConvexos($A, -B$)

Entrada: Polígonos convexos sem buraco A e $-B$.
Saída: $S = A \oplus -B = \text{NFP}_{AB}$.

Início

- Liste as arestas de A em uma ordem angular, referente à inclinação de suas arestas; faça o mesmo para as arestas do polígono $-B$;
- Intercale estas arestas em uma lista F, mantendo a ordem de inclinação;
- Crie o polígono S com a sequência das arestas de F;
- **Retorne** S;

Fim

Diagrama de inclinação

Para realizar o cálculo da AB entre dois polígonos convexos, foram utilizados três tipos de informações das arestas:

- inclinação (em relação a um sistema de coordenadas);
- medida geométrica (comprimento das arestas);
- informação topológica (conexão entre arestas).

Há algumas considerações a serem feitas em relação à estas informações: quando consideramos um vértice do polígono, este não terá comprimento, mas poderá ser representado por um número infinito de inclinações; quando consideramos uma aresta, esta terá comprimento definido e possuirá apenas uma inclinação, que é a inclinação em relação ao eixo das abscissas. Diante disso, podemos tratar estas informações da seguinte forma:

- A inclinação de cada aresta pode ser representada por um ponto em um círculo de raio unitário, enquanto a inclinação de um vértice seria representada pelo arco deste círculo. Nem toda extensão do arco representa uma inclinação possível para um vértice. Podemos restringir estas inclinações àquelas entre arestas adjacentes e que são externas ao respectivo polígono;
- O comprimento de uma aresta não é representado no círculo unitário, mas é uma informação associada ao ponto do círculo que representa a sua inclinação;
- A informação topológica de um polígono convexo está representada no círculo unitário, que é obtida ao seguir a sequência de pontos que representam a inclinação das arestas do polígono. Note que, para obter tal sequência, devemos percorrer o caminho

que representa o círculo em uma única orientação (horária ou anti-horária). Já para o caso de polígonos não convexos, o percurso deste caminho não seguirá uma única orientação, pelo fato da variação dos valores de inclinação entre arestas adjacentes não possuir um comportamento monotônico. Assim, deve ser observada a orientação do percurso sobre o círculo unitário para que se mantenha a conectividade topológica, em que as trocas de orientações representariam elementos (vértices e/ou arestas) não convexos.

Além de uma representação numérica modelada com uma estrutura de dados própria, tais informações podem ser ilustradas em um diagrama, que é chamado *diagrama de inclinação*. O diagrama de inclinação de um polígono convexo A é apresentado na Figura 4.4. Já a Figura 4.5 mostra o diagrama modelado para um polígono não convexo com apenas uma concavidade, evidenciando a alteração na orientação do percurso sobre o círculo unitário durante a análise topológica, de modo que haverá pelo menos uma determinada porção do diagrama em que o caminho será percorrido mais de uma vez nesta análise, sendo uma vez a mais na orientação do polígono.

De posse dos diagramas de inclinação dos polígonos convexos A e $-B$, o cálculo da AB pode ser feito da seguinte forma:

- calcule o diagrama de inclinação do polígono A e do polígono $-B$;
- intercale estes em um único diagrama D;
- realize a concatenação das arestas, segundo a ordem angular que estas aparecem em D, a fim de gerar o polígono resultante S.

Esse processo é apresentado na Figura 4.6. Note que esses passos são idênticos aos seguidos pelo Algoritmo 1, porém com uso de diagramas de inclinação em vez de polígonos.

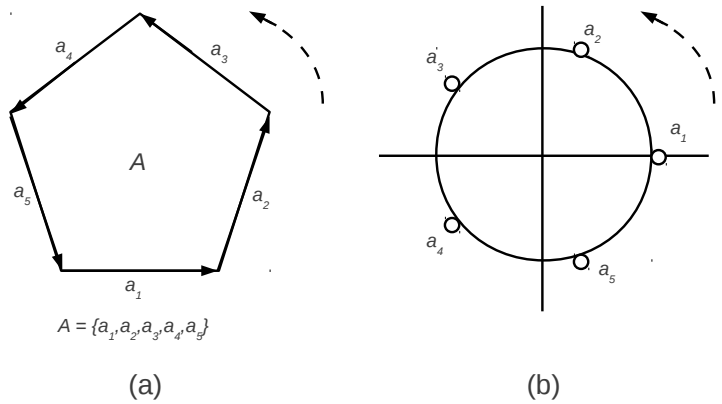


Figura 4.4: Representação de um (a) polígono convexo A e seu (b) diagrama de inclinação.

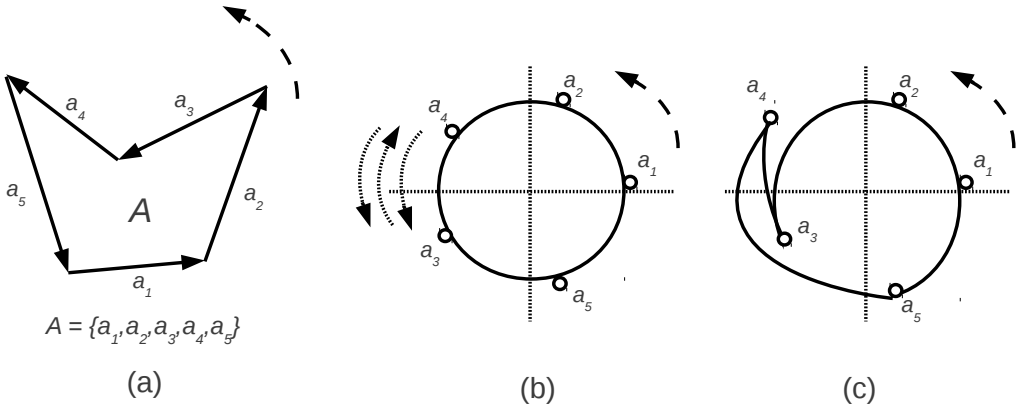


Figura 4.5: Diagrama de inclinação de um polígono simples não convexo: (a) polígono simples não convexo com orientação anti-horária; (b) diagrama de inclinação sobre o círculo unitário; (c) caminho real percorrido sobre o diagrama de inclinação.

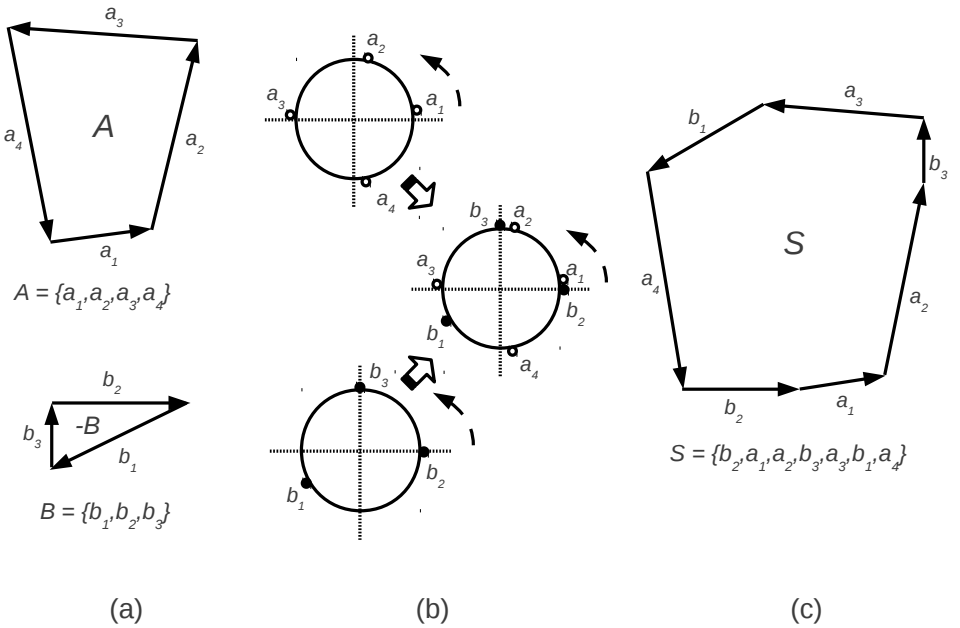


Figura 4.6: Cálculo da Adição de Borda para geração do polígono S: (a) polígonos convexos A e $-B$, e seus respectivos diagramas de inclinação; (b) intercalação dos diagramas de inclinação de A e de $-B$; (c) polígono S, resultante da concatenação das arestas dos polígonos iniciais.

A Adição de Borda considerando polígonos simples não convexos

Quando um dos polígonos não é convexo, a complexidade para se calcular a AB entre estes é maior. Nesse caso, é possível realizar o cálculo da AB, mas com uma consequência: não é possível garantir que todos os elementos serão borda de S, no qual $S = A \oplus -B$. A Figura 4.7 apresenta um exemplo desta situação, em que a parte (b) mostra uma região hachurada que é interior ao polígono resultante S, de modo que S torna-se um polígono não simples (polígono com cruzamento entre arestas não consecutivas). Logo, esses casos exigem um esforço maior para se determinar a borda real de S.

A Adição de Borda entre polígono simples e polígono convexo

Os casos em que apenas um dos polígonos é convexo, o NFP pode ser computado seguindo os passos para a AB do polígono convexo com as partes convexas do outro polígono. Dessa forma, a AB poderá ser computada parcialmente, restando, então, tratar as partes não convexas envolvidas do polígono simples. A Figura 4.8 guiará o entendimento deste processo para a intercalação dos diagramas de inclinação de A e $-B$.

Para se realizar esta intercalação, é necessário que a ordem topológica de ambos os polígonos esteja representada no diagrama de inclinação resultante. Enquanto há convexidade no diagrama de A, a intercalação é feita normalmente. Por exemplo, as porções de b_2 a b_4 de $-B$, em orientação anti-horária, são intercaladas com as arestas envolvidas de A,

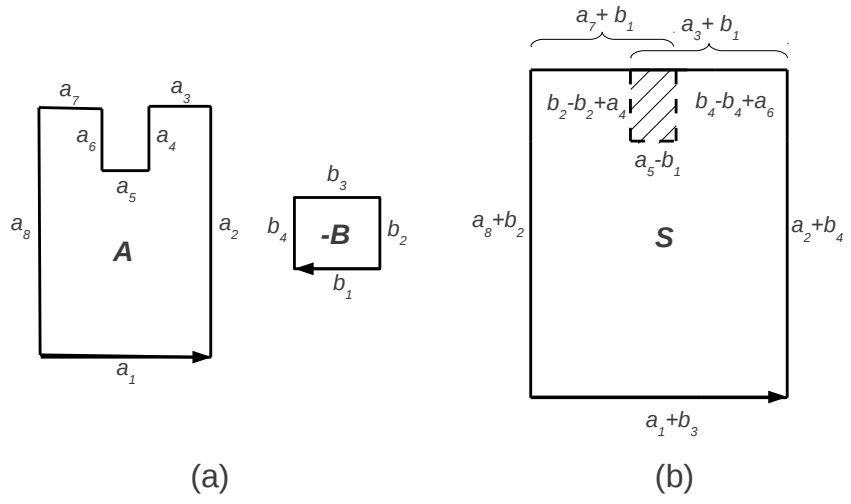


Figura 4.7: Adição de Borda para polígonos simples: (a) polígono não convexo A e polígono convexo $-B$; e (b) polígono resultante S, que possui arestas interiores não pertencentes à borda de S.

como no caso convexo. A partir da inclinação da aresta b_4 no diagrama de $-B$, teremos a alteração das intercalações, devido à parte não convexa de A. Levando em conta este fato, a aresta b_1 encontra-se nessa porção, e, durante o percurso, será considerada três vezes.

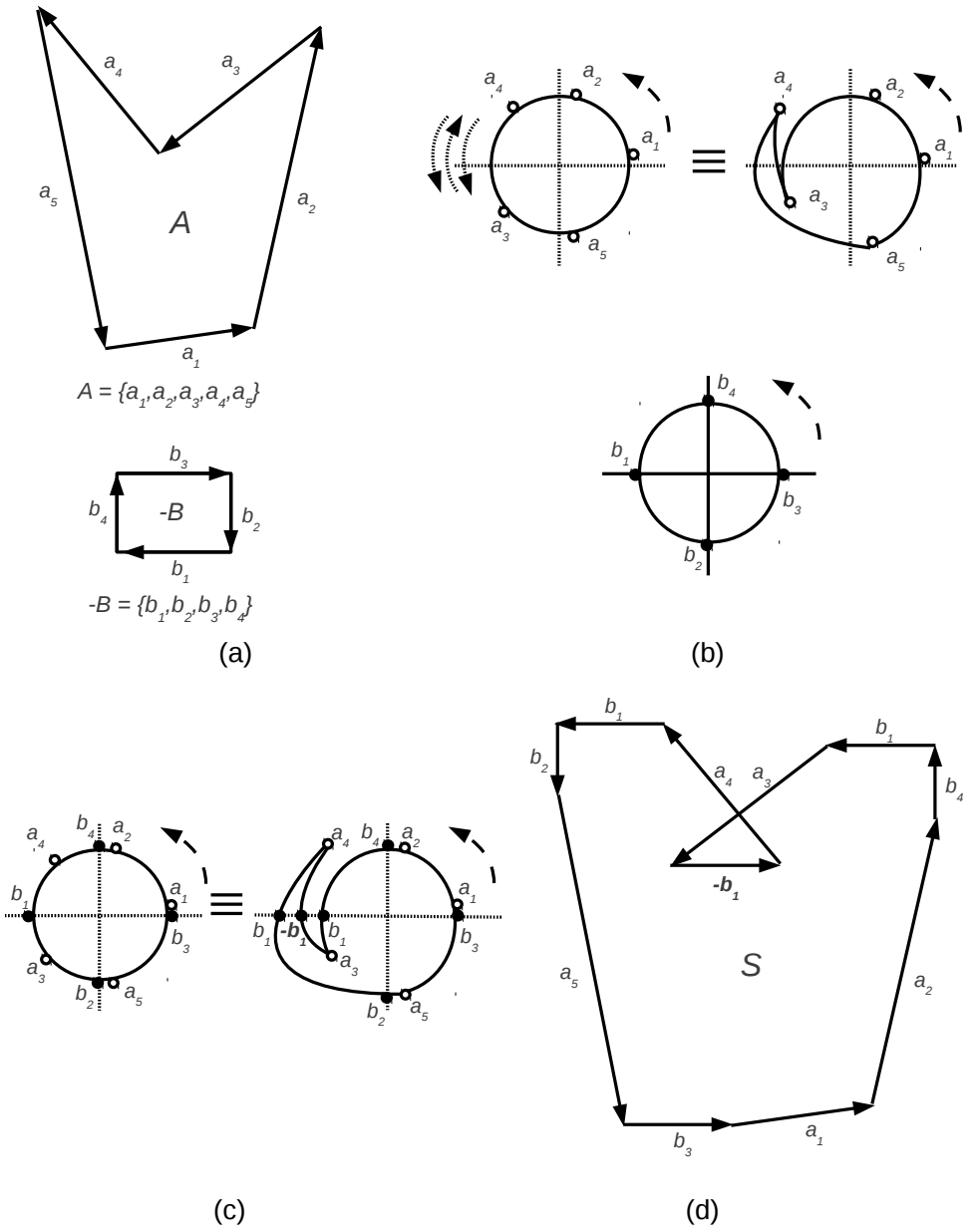


Figura 4.8: Adição de Borda para o caso simples-convexo: (a) polígonos A (simples) e $-B$ (convexo); (b) diagramas de inclinação de A (superior) e de $-B$ (inferior); (c) intercalação dos diagramas de inclinação; e (d) polígono resultante S, composto de arestas interiores à borda de S.

A primeira em orientação anti-horária (orientação do polígono A), fazendo com que esta tenha sua orientação e sentido mantidos, pois representa o seu percurso sobre o vértice convexo da aresta a_3 (a_3 é intercalada, em seguida, no diagrama resultante). A segunda (de a_3 para a_4), em orientação horária, indicando que os elementos de A são não convexos. Nesse caso, deve-se inverter sua orientação e o seu sentido (essa inversão é representada com um símbolo '$-$'), pois representa o percurso desta aresta em orientação oposta, sobre o vértice não convexo entre as arestas a_3 e a_4 (a_4 é intercalada, em seguida, no diagrama resultante). Por fim, a terceira é semelhante à primeira, fazendo com que seja considerada em sua orientação e sentido normais, pois representa o seu percurso sobre o vértice convexo entre as arestas a_4 e a_5 (a_5 é intercalada, em seguida, no diagrama resultante). Feito isso, o diagrama resultante está completo, e a sequência de arestas obtidas neste percurso (percurso este que é semelhante ao do diagrama de A) corresponde à AB de A e $-B$.

É importante notar que a inversão de sentido e orientação de uma aresta só acontece no diagrama que representa o polígono convexo, pois a ordem topológica das arestas do polígono não convexo é sempre seguida de forma sequencial. Outra consideração se refere ao caminho adotado durante a análise dos diagramas de inclinação, que deverá ser aquele representado pelo diagrama referente ao polígono simples não convexo, de modo que todas as arestas possam ser intercaladas em sua posição correta.

Realizada a concatenação das arestas após ser determinada a ordem de ocorrência de cada uma, teremos a geração de um polígono não simples (devido às arestas internas cruzadas). Em seguida, é necessária uma etapa para remoção destas arestas (a explicação de um método para remoção de arestas internas é feita quando tratamos o caso simples-simples), para que o polígono gerado corresponda à SM.

A Adição de Borda de polígonos simples

Como neste caso não há uma técnica que seja a melhor opção para a realização da AB, vamos apresentar a técnica proposta por Bennell e Song [5], que é bastante eficiente e foi utilizada neste trabalho. O objetivo, neste caso, é gerar a menor quantidade de arestas internas, de modo a reduzir o esforço de se determinar quais arestas (ou porções destas) constituirão o polígono resultante que representa o NFP.

A proposta apresentada por Bennell e Song teve sua concepção inspirada no caso da AB para o caso simples-convexo, no qual os diagramas de inclinação podem ser intercalados de forma simples: dados os polígonos A e $-B$, a ideia básica é dividir $-B$ em cadeias de arestas, formando grupos "convexos" e "não convexos", de modo que tais grupos poderão ter seus respectivos diagramas de inclinação intercalados com o diagrama de inclinação de A sem causar conflitos ou ambiguidades na escolha das arestas, o que seria comum caso

trabalhássemos com apenas um diagrama de inclinação de $-B$.

Para entendimento da técnica, apresentamos os passos executados pelo referido algoritmo para o cálculo da AB, considerando os polígonos simples A e $-B$, que estão ilustrados na Figura 4.9.

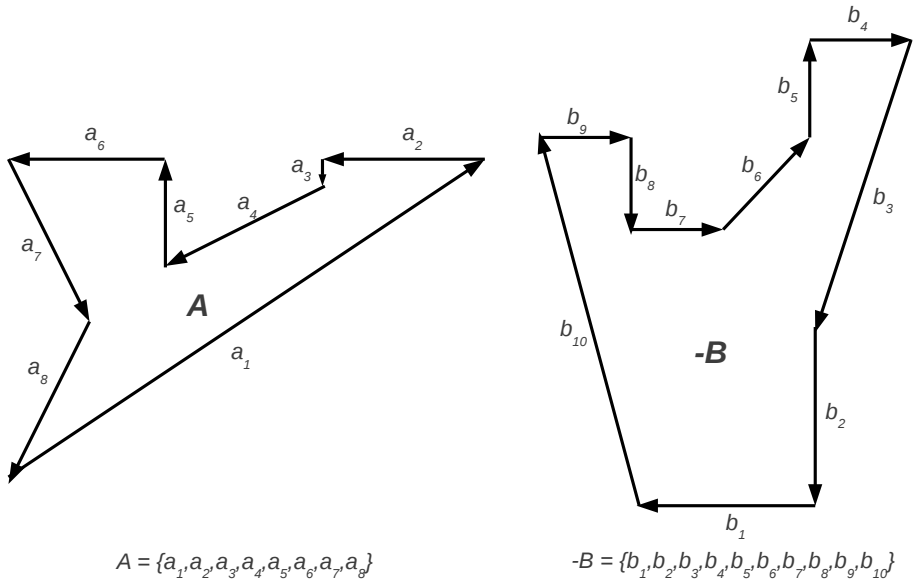


Figura 4.9: Polígonos simples A e $-B$, respectivamente.

Divisão de $-B$ em sequências convexas e não convexas

O primeiro passo consiste em dividir as sequências das linhas poligonais de $-B$ em grupos "convexos" e "não convexos".

A divisão de $-B$ busca encontrar, preferencialmente, grupos convexos, ou seja, as sequências de linhas poligonais ligadas à vertices convexos. Veja que esse é um padrão seguido por Bennell e Song em [5], sendo, também, o padrão neste trabalho. A Figura 4.10 apresenta a divisão dos grupos de $-B$, sendo dois convexos e um não convexo. Os grupos são os seguintes:

- Grupo 1 (convexo): $b_8, b_9, b_{10}, b_1, b_2$;
- Grupo 2 (convexo): b_3, b_4, b_5;
- Grupo 3 (não convexo): b_6, b_7.

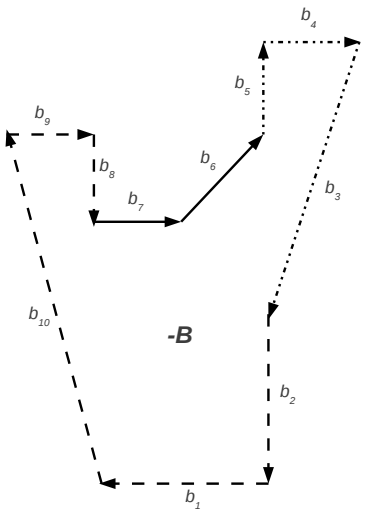


Figura 4.10: Polígono $-B$ representado pela divisão dos grupos convexos e não convexos.

Essa divisão nos permite tratar esse caso de forma parecida com o caso simples-convexo.

Intercalação dos diagramas de inclinação

Considerando, separadamente, cada um dos grupos criados de $-B$, gera-se o respectivo diagrama de inclinação e, em seguida, realiza-se a intercalação com o diagrama de inclinação que representa A. A Figura 4.11 apresenta os polígonos A e $-B$ e seus respectivos diagramas de inclinação, considerando apenas o grupo 1 do polígono $-B$. O resultado desta

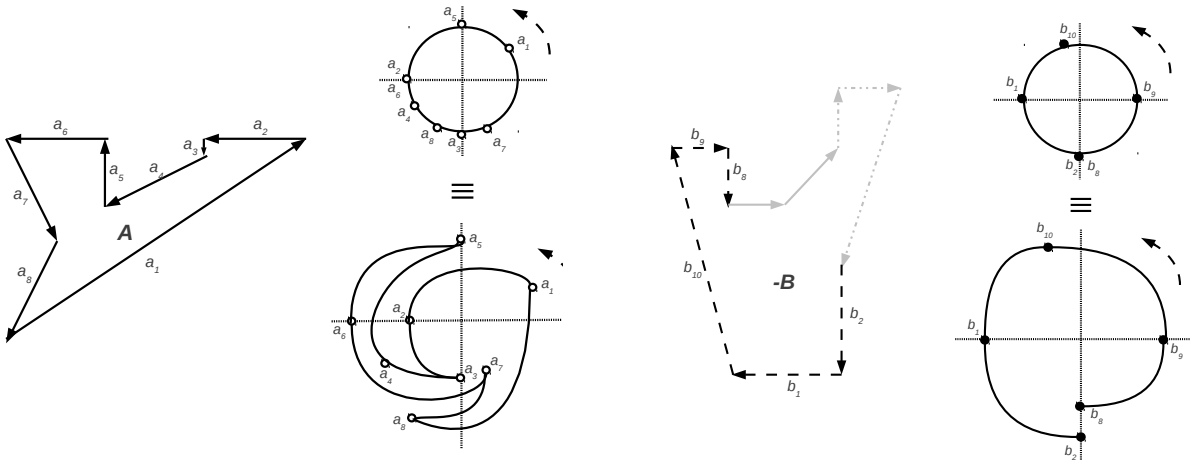


Figura 4.11: Diagramas de inclinação do polígono A e do grupo 1 do polígono $-B$.

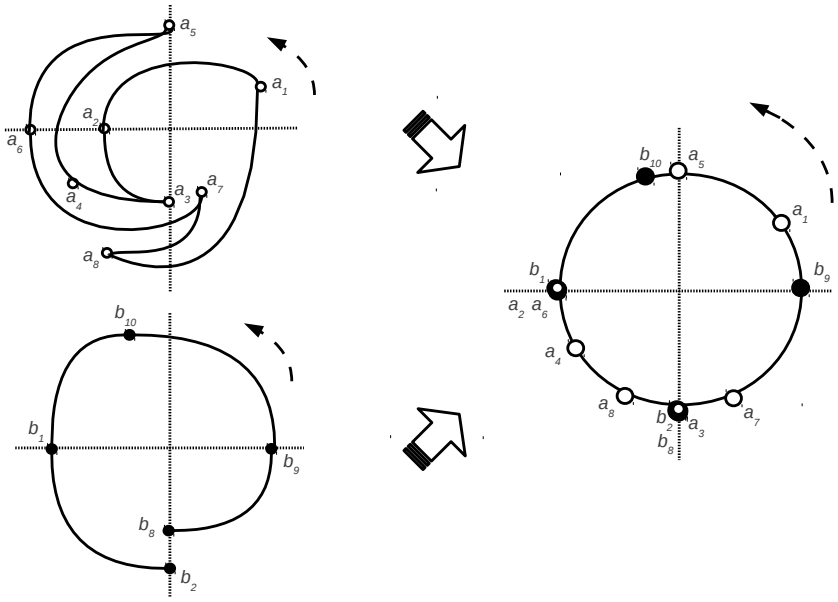


Figura 4.12: Resultado da intercalação dos diagramas de inclinação do polígono A e do grupo 1 do polígono $-B$.

intercalação é apresentado na Figura 4.12.

Abstração do diagrama de inclinação por meio de uma lista

Similar ao caso simples-convexo, deve-se percorrer a ordem topológica das arestas do polígono "não convexo" (devido à criação dos grupos de arestas, assumimos que $-B$ seria "convexo") durante a intercalação dos diagramas de inclinação, aplicando-se as definições vistas. No entanto, a divisão de $-B$ em grupos adiciona uma pequena alteração ao processo, de modo que poderá ser necessário mais de um ciclo completo sobre o diagrama de A (polígono não convexo) para se encontrar a sequência de arestas do diagrama de $-B$ (polígono "convexo"), durante a intercalação destes diagramas. Apesar disso, tal situação é resolvida facilmente com uma lista simples para cada grupo de $-B$, que mantém a ordem de ocorrência das arestas e que indica quais arestas iniciam as porções não convexas dos diagramas intercalados, resumindo os vários ciclos possíveis em uma lista única. Assim, considerando os polígonos A e $-B$ e uma lista L_g para armazenamento da sequência de arestas de um grupo g de $-B$, há dois casos a se considerar na criação dessa lista:

- Grupo de $-B$ é convexo: por padrão, começamos o percurso pela aresta a_1, sendo a orientação anti-horária o fluxo normal de intercalação dos diagramas de inclinação. Cada aresta de A encontrada é inserida em L_g normalmente;

- Grupo de $-B$ não é convexo: por padrão, começamos o percurso pela aresta a_n, sendo n o número de arestas de A, e orientação horária como fluxo normal de intercalação dos diagramas de inclinação. Cada aresta de A encontrada é inserida em L_g com orientação e sentido invertidos.

Em ambos os casos, as arestas de $-B$ incluídas terão orientação e sentido invertidos somente se forem encontradas na porção do diagrama de A que representa uma região não convexa. Caso alguma aresta de A tenha a mesma inclinação de alguma de $-B$, será dada preferência pelas arestas de $-B$ na inserção em L_g. Também, caso alguma aresta de $-B$ tenha a mesma inclinação de uma aresta de A que indica mudança de direção no percurso do diagrama, esta será repetida, porém em orientação e sentido invertidos, após a inclusão da aresta de A. Um exemplo deste caso acontece na intercalação do diagrama de A com o grupo 1 de $-B$, entre as arestas b_2 e a_7 (b_2(orientação anti-horária), a_7(mudança de direção), $-b_2$(orientação horária)).

As listas completas, para cada grupo, são apresentadas a seguir:

- L_1: a_1, b_{10}, b_1, a_2, b_8, b_2, a_3, $-b_2$, $-b_8$, a_4, $-b_1$, $-b_{10}$, a_5, b_{10}, b_1, a_6, b_8, b_2, a_7, $-b_2$, $-b_8$, a_8, b_8, b_2, b_9;
- L_2: $a_1, b_5, a_2, b_3, a_3, -b_3, a_4, -b_5, a_5, b_5, a_6, b_3, a_7, -b_3, a_8, b_3, b_4$;
- L_3: $-a_8, -a_7, -a_6, -a_5, -a_4, -a_3, -a_2, b_6, -a_1, b_7$.

Detecção da sequência de arestas correta

A não convexidade do polígono A pode fazer com que uma dada sequência de arestas L_g, obtida com a intercalação dos diagramas de inclinação, não resulte na sequência correta da AB. Isso porque, com as alterações de inclinação das arestas de determinadas partes da sequência topológica de A, certas arestas topologicamente distantes de um grupo de $-B$ são consideradas durante a intercalação dos respectivos diagramas de inclinação, o que não resultaria na sequência correta. Dessa forma, sabendo que as arestas de A são sempre incluídas na sequência correta durante a análise de L_g, é necessário apenas verificar quando e quais arestas do respectivo grupo de $-B$ também devem ser incluídas.

Para criar a lista final LF_g de cada grupo g de $-B$, que contém a sequência correta de arestas gerada pela AB, é imprescindível que a quantidade de cada aresta do grupo de $-B$ considerado seja previamente determinada. Isso é feito de forma simples, com a contagem de ocorrência destas, enquanto se percorre o diagrama de inclinação de A. Considerando o exemplo para o grupo 1 de $-B$, a aresta b_2 e b_8 serão incluídas em LF_1 cinco vezes, enquanto as arestas b_1 e b_{10} três, e b_9 uma. Em seguida, durante a análise da lista L_g, determina-se a aresta do grupo de $-B$ a ser incluída: dada a primeira aresta

de um grupo de $-B$, a inclusão das próximas arestas deste grupo estará condicionada às chamadas "arestas admissíveis". Uma aresta de um grupo de $-B$ é admissível caso ela mantenha a conectividade da sequência topológica do seu grupo. Desse modo, semelhante ao caso simples-convexo, as arestas de $-B$ devem ser consideradas de forma sequencial, de modo a simular um deslocamento contínuo de $-B$ em torno de todos os vértices de A. Logo, a sequência de arestas admissíveis de $-B$ em L_g estará restrita à duas opções: sua vizinha imediatamente anterior ou sua vizinha imediatamente posterior. Estas opções estão ligadas à direção atual do percurso sobre o diagrama de A, além do tipo de grupo de $-B$ considerado (convexo ou não convexo). Como exemplo, a Figura 4.13 destaca a sequência de arestas admissíveis para o grupo 2 durante a análise de L_2. Nesta figura, os círculos indicam o estado atual da análise, após a inclusão de uma aresta admissível do grupo 2 de $-B$. O círculo denominado "Estado inicial" indica o primeiro estado a se considerar, de modo que a aresta b_3 será a primeira a ser incluída em LF_2. As setas tracejadas representam as transições e indicam as arestas admissíveis durante a análise. Para um grupo não convexo, esta determinação é análoga. Note que as arestas das extremidades de um grupo de $-B$ nunca serão admissíveis ao mesmo instante, visto que quaisquer destes grupos nunca formam um ciclo.

Por fim, considerando estas observações, realiza-se o percurso sobre a lista L_g, iniciando pela primeira aresta do grupo de $-B$ em questão. Caso a primeira aresta do grupo de $-B$ ocorra mais de uma vez em L_g, é escolhida aquela que gera um ciclo correto sobre

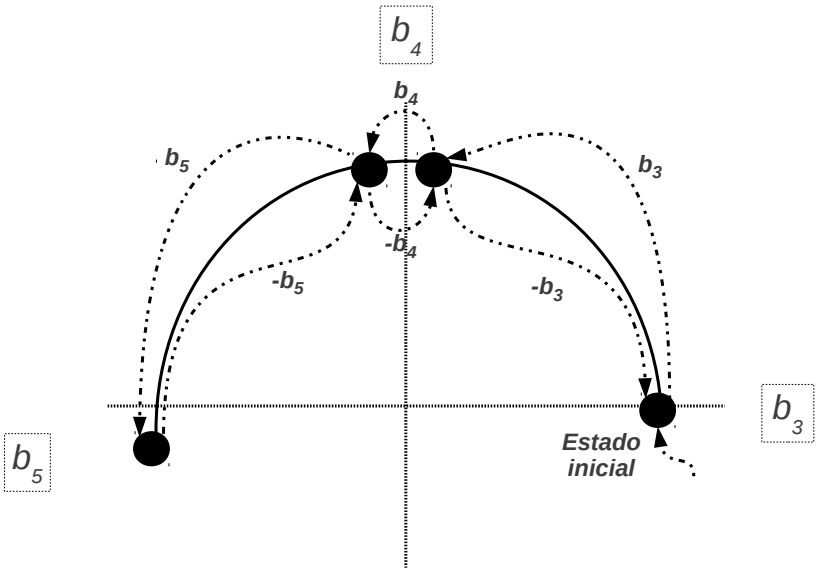


Figura 4.13: Sequência de arestas admissíveis do grupo 2 de $-B$ na análise de L_2.

o diagrama intercalado, ou seja, aquela que permitirá que todas as arestas de seu grupo sejam consideradas, durante a análise de L_g. Levando em conta os três grupos de $-B$, temos as seguintes listas finais:

- LF_1: b_8, a_3, $-b_8$, a_4, a_5, a_6, b_8, a_7, $-b_8$, a_8, b_8, b_9, a_1, b_{10}, b_1, a_2, b_2, a_3, $-b_2$, a_4, $-b_1$, $-b_{10}$, a_5, b_{10}, b_1, a_6, b_2, a_7, $-b_2$, a_8, b_2;
- LF_2: b_3, a_3, $-b_3$, a_4, a_5, a_6, b_3, a_7, $-b_3$, a_8, b_3, b_4, a_1, b_5, a_2, a_3, a_4, $-b_5$, a_5, b_5;
- LF_3: $b_6, -a_1, b_7$

Note que, para os grupos 1 e 2, foram necessárias mais voltas sobre o diagrama de A (ou seja, percorremos mais de uma vez L_1 e L_2), além de que, após uma aresta de um grupo de $-B$ ser incluída, esta é desconsiderada nas próximas voltas.

Concatenação das listas completas

Dadas as listas finais LF_g de cada grupo g de $-B$, a borda final que representa a AB entre A e $-B$ é obtida com a ligação entre tais listas subsequentes (ou seja, $LF_1 \rightsquigarrow LF_2 \rightsquigarrow LF_3 \rightsquigarrow \ldots \rightsquigarrow LF_n \rightsquigarrow LF_1$, em que "$\rightsquigarrow$" representa uma ligação entre duas listas), considerando um fato importante: como o polígono $-B$ foi dividido, a última aresta de A de uma lista final LF_g pode ser diferente da primeira aresta de A de uma lista final LF_{g+1}, de modo que a ligação destas, nesse caso, formaria uma borda final incorreta. Logo, para que as listas consecutivas possam ser adequadamente concatenadas, deve-se ajustá-las, de modo que se mantenha a ordem topológica de A entre as listas finais.

Diante disso, o ajuste é feito em cada lista final LF_g, incluindo nestas algumas arestas de A. Por padrão, adicionamos tais arestas sempre ao final das listas. Note que aqui temos uma adição exclusiva de arestas de A, de modo que a adição destas em LF_g é baseada em sua sequência topológica e no tipo do grupo de $-B$ considerado ("convexo" ou "não convexo"). Assim, considerando a última aresta de A incluída na lista LF_g, a primeira aresta de A incluída na lista LF_{g+1} e o tipo de grupo de $-B$ que gera a LF_g, a adição de arestas de A, chamadas de "arestas de ligação", será baseada na seguinte condição:

- Grupo de $-B$ de LF_g é convexo: serão adicionadas arestas inversas da sequência topológica de A, até a posição que corresponde a primeira aresta de A do grupo posterior (e incluí-la se os sentidos forem opostos);
- Grupo de $-B$ de LF_g não é convexo: serão adicionadas arestas da sequência topológica de A, até a posição que corresponde a primeira aresta de A do grupo posterior (e incluí-la se os sentidos forem opostos);

Como exemplo, os grupos LF_1 e LF_2 terão a seguinte ligação: a última aresta de A em LF_1 é a a_8, enquanto a primeira de A em LF_2 é a a_3. Com base na condição anterior, a ligação é feita adicionando-se a seguinte sequência de arestas: $-a_8$, $-a_7$, $-a_6$, $-a_5$, $-a_4$ e $-a_3$. Feito isso, ambas as listas estão sobre a mesma porção de A e podem ser concatenadas. Note que, se algum grupo não possuir pelo menos uma aresta de A, esse procedimento não chegará a um resultado viável. Logo, nos casos em que isso acontece, basta incluir a aresta de A que precede a aresta de $-B$ durante o percurso sobre o diagrama de inclinação de A. A seguir, apresentamos as arestas que correspondem à borda do polígono resultante, destacando as suas arestas de ligação (sublinhadas), separadas pelos grupos de $-B$:

- Grupo 1: b_8, a_3, $-b_8$, a_4, a_5, a_6, b_8, a_7, $-b_8$, a_8, b_8, b_9, a_1, b_{10}, b_1, a_2, b_2, a_3, $-b_2$, a_4, $-b_1$, $-b_{10}$, a_5, b_{10}, b_1, a_6, b_2, a_7, $-b_2, a_8, b_2$, $\underline{-a_8, -a_7, -a_6, -a_5, -a_4, -a_3}$;
- Grupo 2: b_3, $a_3, -b_3, a_4, a_5, a_6, b_3, a_7$, $-b_3$, a_8, b_3, b_4, a_1, b_5, a_2, a_3, a_4, $-b_5$, a_5, b_5,$\underline{-a_5, -a_4, -a_3, -a_2}$;
- Grupo 3: $b_6, -a_1, b_7$, $\underline{a_1, a_2}$.

Ao final, seguindo a ordem dos grupos formados, cada uma destas listas deverá ser concatenada, a fim de se gerar o polígono não simples resultante da AB, apresentado na Figura 4.14.

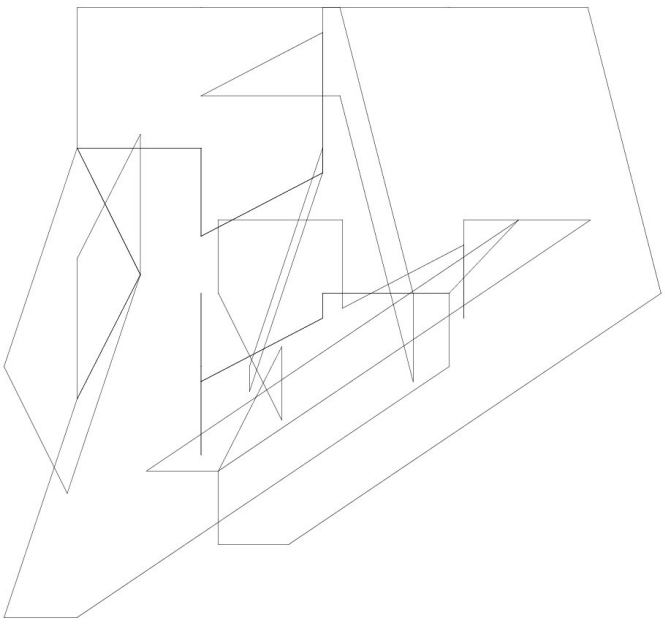


Figura 4.14: Polígono não simples formado pela AB entre os polígonos A e $-B$.

Formação da borda real do $\mathbf{NFP}_{AB}$

Dado um polígono não simples resultante da AB entre A e $-B$, a tarefa agora é retirar as arestas internas que não fazem parte de sua borda real, de modo a torná-lo simples e, assim, represente o correto NFP.

Durante esse processo, há dois casos em que a remoção de arestas pode ser feita diretamente: a borda do polígono resultante segue a orientação anti-horária, logo, arestas com orientação horária podem ser excluídas; e as arestas de ligação das listas (passo anterior) servem apenas para posicionar corretamente as listas finais consecutivas durante a concatenação, de modo que, também, podem ser retiradas. Terminada essa remoção, o polígono obtido será desfeito, formando um conjunto de segmentos parcialmente ligados, denominados "cadeias poligonais", apresentados na Figura 4.15.

A verificação das arestas reais da borda do polígono resultante será feita com base na direção de intersecção entre as cadeias poligonais, pelo método introduzido por Ramkumar [38], que é resumido a seguir.

Dado um conjunto de cadeias poligonais, a ideia é analisar as intersecções entre segmentos de cada uma. São analisadas as possíveis intersecções de seus segmentos com outros segmentos de sua própria cadeia (desde que não sejam segmentos consecutivos), além das possíveis intersecções com segmentos de outras cadeias. Por termos, por padrão,

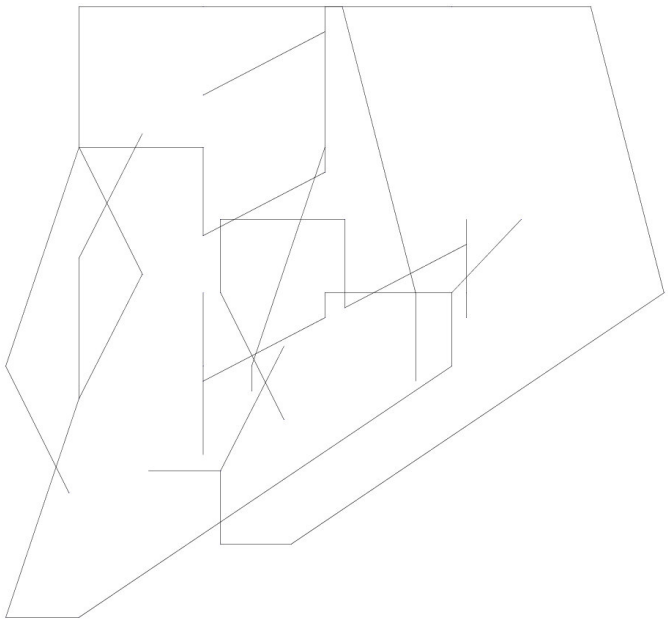


Figura 4.15: Cadeias poligonais obtidas pela remoção das arestas negativas e de ligação do polígono não simples, gerado pela AB entre os polígonos A e $-B$.

uma orientação anti-horária das arestas, as intersecções ocorridas da esquerda para a direita entre um segmento da cadeia considerada e um outro segmento indicam que a cadeia em questão está saindo de uma região interna, e, desse modo, pode representar alguma das bordas externas do polígono resultante. Quando isso acontece, marcamos tal intersecção com um '−'. De modo inverso, as intersecções ocorridas da direita para a esquerda entre um segmento da cadeia considerada e um outro segmentos indicam que a cadeia em questão está entrando em uma região interna, e, desse modo, pode representar o fim de alguma das bordas externas do polígono resultante. Quando isso acontece, marcamos tal intersecção com um '+'. Feita essa análise para uma cadeia poligonal, são mantidos apenas os segmentos (ou porções destes) que estão dentro dos intervalos '−' à '+', por serem candidatos à borda do NFP. Um exemplo desta análise é apresentado na Figura 4.16. Note que este processo deve ser feito, separadamente, para cada uma das cadeias poligonais formadas.

Note que, arestas coincidentes e pontos interiores –seção 4.3.1– serão mantidos pelo método de análise descrito, pois os segmentos para estes elementos ocorrem como pares de arestas com sentidos opostos (ou diferentes) que se interseptam em segmentos não consecutivos de uma cadeia poligonal. Estes casos podem ser vistos na Figura 4.17.

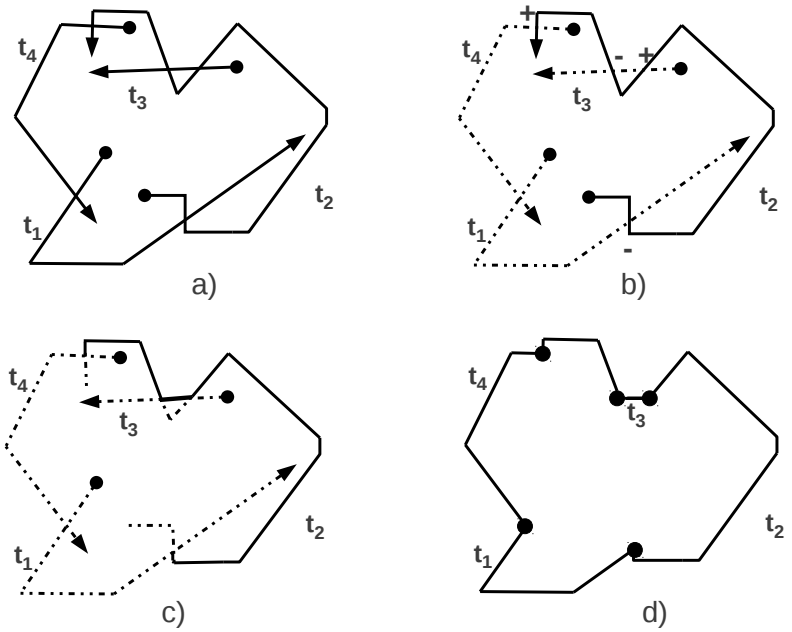


Figura 4.16: Determinação de possíveis bordas para a geração de um polígono: a) cadeias poligonais geradas; b) análise da cadeia poligonal t_2; c) porções da cadeia poligonal t_2 entre os sinais − e +; d) borda completa do polígono resultante.

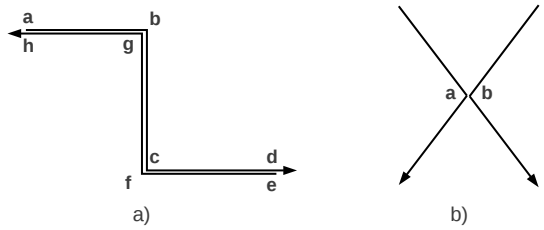


Figura 4.17: Disposição das cadeias poligonais para formação das: a) arestas coincidentes; e b) ponto interior.

Ao final, os segmentos resultantes deverão ser concatenados, de modo que um ciclo é gerado, formando, novamente, um polígono. No entanto, alguns ciclos internos que não representam arestas da borda do NFP podem ter sido formados. Assim, o último passo do algoritmo corresponde à verificação da viabilidade de tais ciclos. Isso é feito analisando-se a disposição entre os polígonos A e B, com o uso do polígono resultante, de forma similar à descrita na seção 4.3. Logo, caso algum ciclo interno gere sobreposição entre A e B, este ciclo será descartado do polígono que representa o NFP_{AB}.

Resumo do algoritmo e análise de complexidade

O resumo do procedimento de AB para polígonos simples sem buraco para obtenção da borda externa do NFP dos polígonos A e $-B$ e a descrição da complexidade dos métodos implementados são apresentados pelo Algoritmo 2.

Algoritmo 2: AdicaoDeBorda_PoligonosSimples($A, -B$)

Entrada: Polígonos simples sem buraco A (com m arestas) e $-B$ (com n arestas).
Saída: $S = A \oplus -B = \text{NFP}_{AB}$.

Início

```
Bgrupos = Divide(B);                                   /* O(n) */
ADiagrama = GeraDiagrama(A);                           /* O(m log m) */
Para cada b ∈ Bgrupos faça                             /* O(mn) */
    BDiagrama ← GeraDiagrama(b);
    merg ← Intercala(ADiagrama, BDiagrama);
    seq ← seq ∪ SeqCorreta(merg);
LigaGrupos(seq);                                       /* O(mn) */
cadeias ← GeraCadeiasPoligonais(seq);                  /* O(mn) */
Para cada c1 ∈ cadeias faça                            /* O((m²n²)²) */
    Para cada c2 ∈ cadeias faça
        bordas ← bordas ∪ BuscaBorda(c1, c2);
S ← ligaBordas(bordas);                                /* O((m²n²)²) */
verificaCiclos(S);                                     /* O((m²n²)²) */
Retorne S;
```

Fim

Capítulo 5

Heurísticas

Este capítulo apresenta as novas heurísticas de empacotamento propostas no trabalho. Estão divididas em heurística básica –composta por uma heurística de agrupamento e uma heurística baseada em Algoritmo Genético– e heurística adaptada –composta pela integração das heurísticas básicas–. As estruturas utilizadas para a otimização do tempo de execução dos algoritmos também são descritas.

5.1 Introdução

Devido à dificuldade de se trabalhar com itens de formatos não simples (e.g., formatos diferentes de círculos, retângulos, triângulos), a maior parte das soluções geradas para os problemas de empacotamento são soluções heurísticas, quando consideramos esta classe de instância. Dentre as heurísticas que encontramos na literatura, há a divisão em duas categorias:

- Abordagem construtiva: constroem soluções adicionando um item por vez e mantendo a viabilidade da solução, ou seja, sem sobreposição entre itens e todos os itens dentro do recipiente;

- Redução de sobreposição: começando com um leiaute não necessariamente válido, a heurística tenta encontrar uma solução válida minimizando uma função de custo, que leva em consideração a inviabilidade da solução.

As heurísticas propostas para os problemas de empacotamento considerados neste trabalho pertencem a abordagem construtiva. A seguir, apresentamos as estruturas básicas necessárias para o desenvolvimento de tais heurísticas.

5.2 Heurísticas elementares

As heurísticas propostas têm em comum algumas heurísticas elementares para a geração de soluções para os problemas. Dentre elas, temos a heurística de agrupamento dos itens, denominada "Heurística de Agrupamento" (HA), e a heurística de ordenação do vetor de itens do empacotamento, baseada na meta-heurística "Algoritmo Genético" (AG), e são apresentadas a seguir.

5.2.1 Heurística de agrupamento

Com o objetivo de criar um agrupamento dos itens que seja compacto, a utilização de propriedades geométricas dos itens durante a construção de uma solução se mostra como uma boa alternativa, sendo considerada em trabalhos com bons resultados práticos (Oliveira *et al.* [36], Bennell e Song [6], Sato *et al.* [39]). Seguindo esta vertente, a HA faz uso da busca estendida proposta por Adamowicz e Albano [1].

Na busca estendida, dados dois polígonos simples P e Q e o NFP_{PQ}, a ideia é encontrar um ponto sobre a borda de NFP_{PQ} tal que, quando Q é empacotado neste ponto, a envoltória retangular cobrindo P e Q tem área mínima. A Figura 5.1 apresenta os passos tomados pela busca estendida para analisar e escolher um agrupamento de itens, considerando as respectivas envoltórias retangulares geradas. Dados o empacotamento parcial e um item em 5.1 a), a busca estendida encontra a posição em que estes itens são agrupados e geram a envoltória retangular de área mínimia (5.1 b) e 5.1 c), respectivamente). Tomando os dois agrupamentos gerados e avaliando a área da respectiva envoltória retangular de cada um, nota-se que 5.1 c) gera uma área retangular menor, representando o agrupamento escolhido. No entanto, existem casos em que o agrupamento produzido pela busca estendida não representa a envoltória retangular de menor área.

A HA considera que os itens a serem empacotados estão em uma lista em alguma ordem pré-estabelecida. O objetivo é remover um item da lista e empacotá-lo no recipiente, considerando um empacotamento parcial D de itens previamente empacotados. A cada iteração, a HA seleciona um novo item Q e testa, utilizando a busca estendida, o empacotamento de Q em D, em cada uma de suas rotações. O agrupamento da busca estendida com envoltória retangular de menor área é então utilizado e passa-se à avaliação de um novo item. Note que a ordenação da lista de itens a serem empacotados influencia na solução da HA. Isto é a chave do Algoritmo Genético que implementamos, já que uma

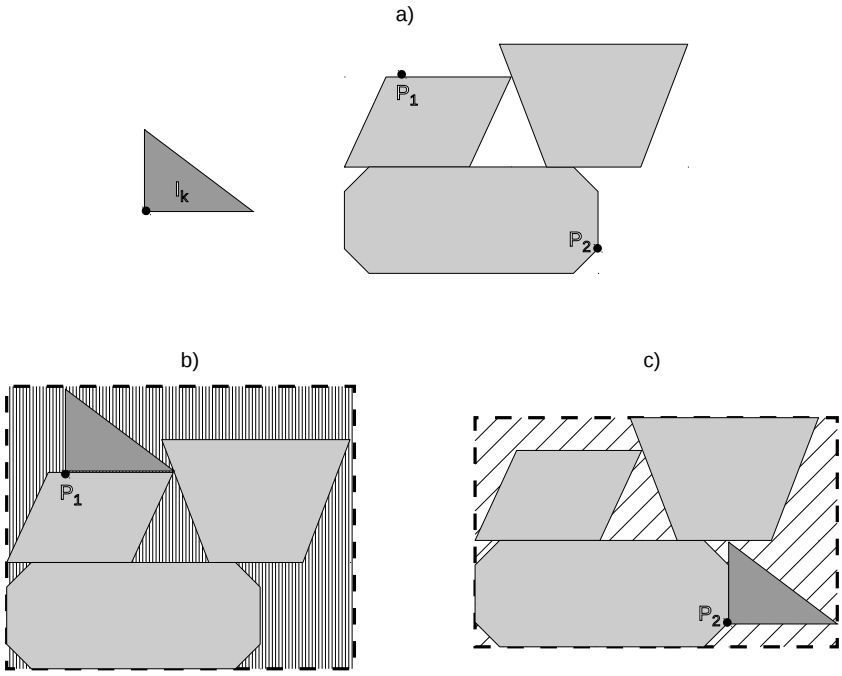


Figura 5.1: Representação da busca estendida considerando apenas dois pontos (P_1 e P_2) de empacotamento do item I_k na solução parcial para análise da envoltória retangular: a) item I_k e a solução parcial composta por três itens; b) agrupamento para o item I_k no ponto P_1 e a envoltória retangular obtida; c) agrupamento para o item I_k no ponto P_2, representando a envoltória retangular de área mínima.

solução será sempre caracterizada pela ordem dos itens na lista de entrada.

Por si só, a utilização da HA baseando-se apenas em escolhas gulosas da busca estendida pode levar à geração de agrupamentos finais pouco compactos. A Figura 5.2 retrata um caso em que isso acontece, no qual a instância apresentada em 5.2 a) não seria agrupada de forma a se obter o melhor agrupamento, qualquer que seja a ordem considerada pela HA.

Diante disso, aplicamos duas variações na busca estendida. Em vez de utilizarmos as posições sobre a borda do NFP que a busca estendida analisa para tentar encontrar o agrupamento com envoltória retangular de área mínima, criamos um novo conjunto de posições de teste de duas maneiras diferentes: a primeira, denominada *Busca Discretizada*, escolhe a posição do próximo item com base em um conjunto de pontos de teste distribuídos em espaços equidistantes sobre a borda do respectivo NFP; a segunda, denominada *Busca Aleatória*, escolhe a posição do próximo item com base em um conjunto aleatório de pontos de teste sobre a borda do respectivo NFP. Como estas formas de agrupamento de itens são extensões da busca estendida, a avaliação da posição do item a ser

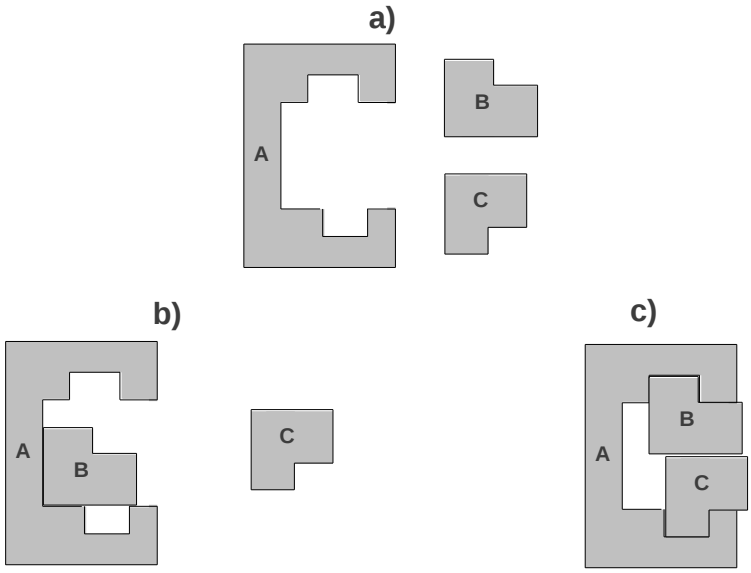


Figura 5.2: Instância em que a solução ótima não pode ser gerada pela HA. a) instância formada pelos itens A, B e C, não sendo permitidas rotações destes; b) provável empacotamento inicial feito pela Heurística de Agrupamento; c) empacotamento ótimo (de melhor aproveitamento).

empacotado ao considerar cada um dos pontos do conjunto de teste poderá ser feita utilizando tanto a área da envoltória retangular (semelhante à avaliação da busca estendida) quanto a área da envoltória convexa obtidas com a solução parcial, o que fornece novas possibilidades para o empacotamento de um item. A Figura 5.3 apresenta um exemplo de um conjunto de pontos de teste, para cada uma destas variações.

Durante o empacotamento de um item Q em uma solução parcial D, a escolha de agrupamentos que geram menor comprimento no recipiente poderá ser mais interessante a agrupamentos mais compactos, mas com comprimento maior, visto que a melhor alternativa para o problema *Strip Packing* é construir uma solução que utilize o menor comprimento possível do recipiente. Diante disso, para decidir em qual ponto empacotar Q em D, utilizamos um critério de escolha baseado em um parâmetro de perda α da seguinte forma: seja o aproveitamento de área de um agrupamento qualquer definido como a razão da área dos itens em D mais Q sobre a área da envoltória (retangular ou convexa) do agrupamento de D com Q. Com base nesse aproveitamento, considere as posições analisadas pela busca estendida (ou alguma de suas variações); seja A_R o valor do aproveitamento empacotando-se Q no ponto que minimiza a área da envoltória retangular com D; e seja A_C o valor do aproveitamento quando empacota-se Q no ponto que minimiza o comprimento do empacotamento resultante com D. Assim, se temos

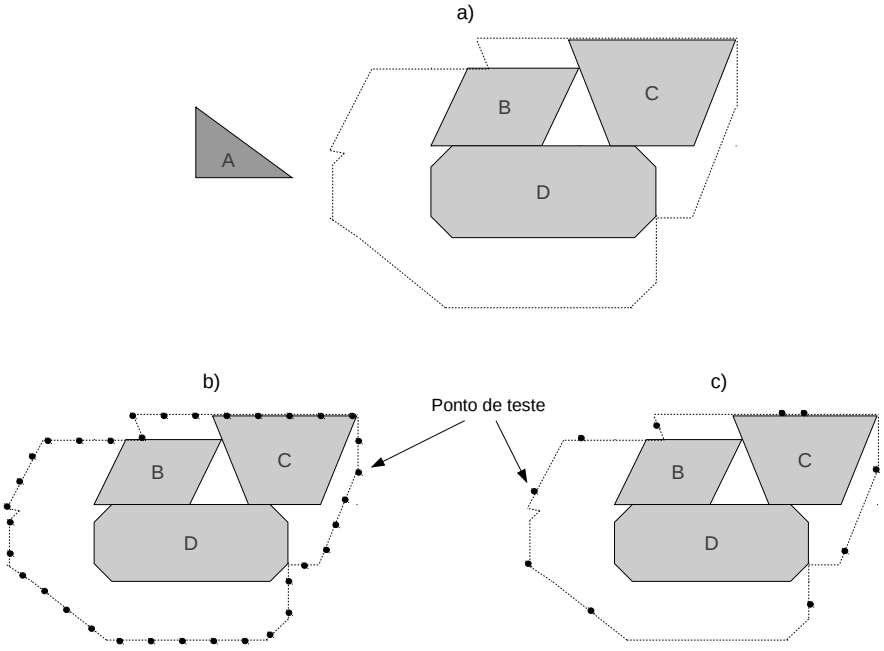


Figura 5.3: Variação da busca estendida, onde: a) A é o próximo item a ser empacotado junto à solução parcial (itens B, C e D), e o NFP entre eles; b) conjunto de pontos de teste para a Busca Discretizada; e c) conjunto de pontos de teste para a Busca Aleatória.

$$A_R - A_C < \alpha, \tag{5.1}$$

será dada preferência ao segundo empacotamento, mesmo tendo aproveitamento menor. Esta escolha durante o empacotamento pode ser vista na Figura 5.4, de modo que os valores de aproveitamento obtidos em 5.4 c) serão utilizados pela Equação 5.1.

Tomando toda a descrição anterior, o Algoritmo 3 apresenta uma síntese dos passos seguidos pela HA proposta para realizar o agrupamento da solução. Note que, para as variações de busca propostas, podemos utilizar a área da envoltória convexa em vez da retangular no cômputo do aproveitamento.

Um ponto a se destacar da heurística descrita pelo Algoritmo 3 é com relação ao método Empacota, que utiliza a estrutura de NFPs para verificar pontos viáveis de empacotamento, considerando a solução parcial. Nesse método, o NFP que é utilizado em tais buscas não será recalculado por qualquer dos algoritmos descritos na seção 4.3, à cada vez que se empacota um novo item, mas será utilizada a união de NFPs dos itens individuais da solução parcial, por questões de eficiência. Note que, nesse caso, a geração de todas as arestas do NFP resultante (arestas de borda, arestas coincidentes e pontos interiores, vistos na seção 4.3) também será dependente do método de união dos polígonos individuais.

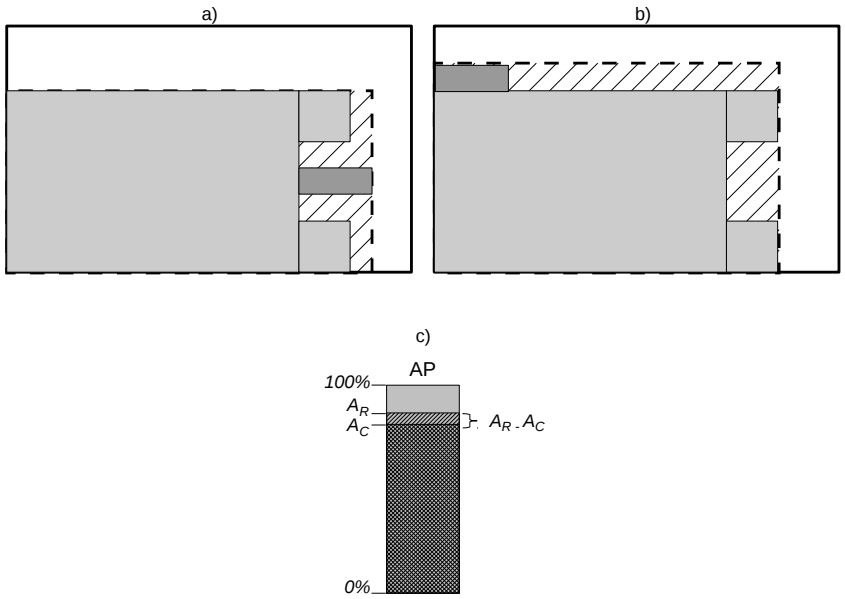


Figura 5.4: Teste de agrupamento segundo o aproveitamento e comprimento da solução; a) empacotamento com melhor aproveitamento AP; b) empacotamento de menor comprimento no recipiente; c) representação do fator de perda dos aproveitamentos anteriores.

5.2.2 Algoritmo Genético

O Algoritmo Genético (AG), proposto por Holland [24], é uma meta-heurística estocástica de busca, baseada em computação evolucionária, que imita o processo de evolução natural, sendo largamente utilizada como método de otimização para problemas difíceis.

Inicialmente, foi concebida como um meio de se estudar o comportamento adaptativo. Por essa característica, originalmente foram apresentados vários operadores desta natureza, como a seleção de indivíduos, a mutação e a recombinação. Para mais detalhes sobre esta técnica, uma discussão acerca de suas características foi feita por Eiben e Smith [18].

A ideia proposta para utilizar o AG nos problemas de empacotamento é aplicá-lo de forma que, para uma determinada instância, gere diferentes ordens da lista de itens que será empacotada pela HA (descrita na seção 5.2.1). A seguir, descrevemos a modelagem de suas estruturas para trabalhar com a heurística de empacotamento proposta.

Indivíduos

Um indivíduo é representado por uma lista dos itens a serem empacotados em uma determinada ordem. Dada esta lista, uma solução é unicamente obtida utilizando-se a HA (seção 5.2.1).

A criação dos indivíduos da população inicial é feita de forma determinística, na qual a

Algoritmo 3: HEURISTICADEAGRUPAMENTO(i, sol, t_busca, $t\_envoltoria$, α, R)

Entrada: i: item a ser empacotado;
sol: solução parcial do problema de empacotamento;
t_busca: tipo de busca para empacotamento do item;
$t\_envoltoria$: tipo de envoltória para cálculo de aproveitamento;
α: parâmetro para escolha final da solução parcial;
R: recipiente em que estará contida a solução parcial criada.

Saída: A melhor solução, segundo o parâmetro de escolha α.

Início

```
pontosdeteste ← GERAPONTOS(sol, i, t_busca);
Para todo p ∈ pontosdeteste faça
    solucaoauxiliar ← EMPACOTA(p, i, sol, R);
    Se t_busca = BUSCA_ESTENDIDA então
        AVALIASOLUCAO(solucaoauxiliar, EnvoltoriaRetangular);
    senão
        AVALIASOLUCAO(solucaoauxiliar, t_envoltoria);
    conjuntodesolucoes ← conjuntodesolucoes ∪ solucaoauxiliar;
Se está sendo utilizado o critério α então
    retorna EscolheSolucao(conjuntodesolucoes, α);
senão
    retorna SolucaoMelhorAP(conjuntodesolucoes);
```

Fim

escolha da sequência de itens é feita mantendo os itens de maior área nas posições iniciais da respectiva lista.

O *fitness* de um indivíduo depende do problema que será resolvido, podendo ser o aproveitamento do recipiente (*Knapsack*), o número de recipientes utilizados (*Bin Packing*) ou o comprimento do empacotamento encontrado pela HA (*Strip Packing*).

Reprodução

A geração de novos indivíduos está inteiramente baseada na mutação dos indivíduos da solução. A mutação é composta pela aplicação de uma única operação de alteração sobre a lista de itens (cromossomo), dentre os quais podem ser utilizados os seguintes operadores sobre os itens (alelos):

- *swap*: troca os alelos entre duas posições aleatórias de um cromossomo;

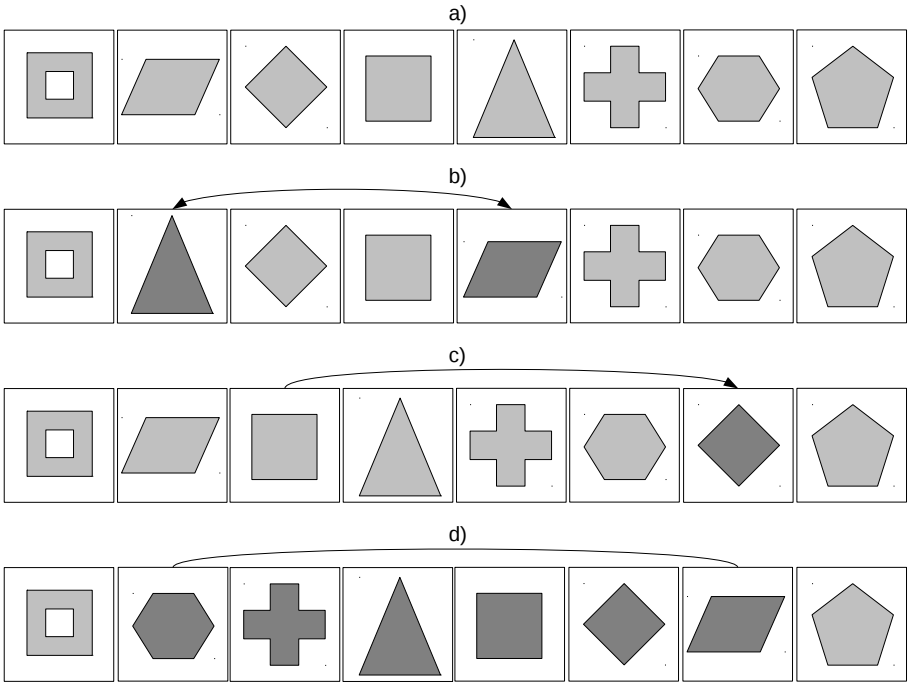


Figura 5.5: Mutações utilizadas sobre o vetor de itens: a) vetor original; b) *swap*; c) inserção; d) inversão.

- *inserção*: retira e reinsere um alelo em posições escolhidas aleatoriamente;

- *inversão*: inverte a ordem dos alelos dentro de um intervalo aleatório do cromossomo.

Um exemplo com aplicação destes operadores está representado na Figura 5.5.

É importante notar que, apesar da recombinação ser uma operação considerada muito importante em qualquer implementação de AG, não o aplicamos sobre os indivíduos pelo fato de não conhecermos uma forma de se aproveitar partes consideradas boas de uma solução, visto que a HA trabalha sobre a solução parcial do empacotamento.

Seleção

A seleção da próxima geração é feita, inicialmente, mantendo-se o melhor indivíduo da população. Em seguida, os demais indivíduos são escolhidos pelo método de torneio: dois indivíduos são escolhidos aleatoriamente, e passa para a próxima geração o melhor indivíduo. Este processo é repetido enquanto o número de indivíduos da próxima geração não estiver completo.

Demais estruturas e operações

Além da modelagem básica do AG descrita anteriormente, outras estruturas de dados são fundamentais para uma eficiente execução do algoritmo. Dentre elas, destacamos as várias estruturas de otimização utilizadas para agilizar o processo de construção de soluções, de modo que se aumente a quantidade de novas soluções geradas e, assim, também as chances de se encontrar soluções mais adaptadas. Entre as estruturas utilizadas, temos:

Cache de soluções

Conjunto de operações sobre um cromossomo, com o objetivo de evitar o recômputo de todo o empacotamento após a mutação do respectivo indivíduo. Funciona da seguinte forma: dado um cromossomo, encontra-se a primeira posição da lista de itens com o alelo que esteja em uma posição diferente, quando comparada com a lista de itens antes da mutação. Considerando esta posição como índice i, o empacotamento com os $i-1$ itens iniciais será mantido (desde que $i > 0$), já que o agrupamento destes na solução é o mesmo.

Cálculo de solução da sequência de mutações

Verificação que proporciona evitar o recômputo da solução de um cromossomo. Nesse caso, quando o cromossomo passa por uma mutação em que a alteração feita gera uma sequência da lista de itens que resulta no mesmo empacotamento, não será aplicada a HA, visto que já existe o valor de *fitness* para este indivíduo.

Restart da população

Reinicialização do AG com nova população. A heurística verifica se há estabilização da solução em mínimo local, e, caso não haja melhorias do *fitness* do melhor indivíduo em um período de tempo determinado, toda a população é reiniciada, realizando-se um embaralhamento na ordem dos itens da lista a ser empacotada de cada indivíduo, a fim de que possam ser geradas novas soluções. Vale ressaltar que o melhor indivíduo gerado até o momento é preservado e reinserido na nova população formada.

Pseudo-código

O Algoritmo 4 resume os passos seguidos pela heurística proposta, baseada em Algoritmo Genético, para buscar soluções de um problema de empacotamento.

Algoritmo 4: HeuristicaBaseadaEmAG($configs$)

Entrada: $configs$: parâmetros de configuração da heurística baseada em AG.
Saída: O indivíduo da população com melhor *fitness*, ou seja, o indivíduo com o melhor empacotamento gerado pela HA.

```
Início
    ConfiguraEstruturas(configs);
    populacao ← cria_populacao();
    CalculaFitness(populacao);
    Enquanto ¬CondicaoDeParada(populacao, TempoAtual()) faça
        Mutacao(populacao);
        CalculaFitness(populacao);
        Selecao(populacao);
        Se solucao_estagnada() então
            Restart(populacao);
            ultima_melhoria_solucao = TempoAtual();
    melhor_individuo ← ObtemMelhor(populacao);
    Retorna melhor_individuo;
Fim
```

5.3 Heurísticas propostas

Com base nas heurísticas elementares da seção anterior, propomos as heurísticas de empacotamento para os problemas considerados neste trabalho. Descrevemos cada uma destas nas próximas seções.

5.3.1 Heurísticas básicas

As primeiras heurísticas propostas para os três problemas de empacotamento, denominadas heurísticas básicas, têm como base as heurísticas descritas na seção 5.2. Dessa forma, a ordenação do vetor de itens a serem empacotados é realizada pelo AG, enquanto o empacotamento de um item no recipiente é feito pela HA.

Como cada problema considerado tem uma definição específica, explicitamos os detalhes de cada uma das heurísticas básicas a seguir.

Strip Packing e *Knapsack*

Constituem as heurísticas no qual existe apenas um único recipiente disponível durante o processo de empacotamento. Para os problemas *Strip Packing* e *Knapsack*, a heurística proposta tem como base as heurísticas elementares para os problemas de empacotamento (seção 5.2). Porém, para o *Knapsack*, há uma pequena alteração: devido o comprimento fixo do recipiente, quando o próximo item da lista não puder ser empacotado, tal item será descartado da solução.

Bin Packing

Semelhante aos problemas anteriores, também se baseia nas heurísticas elementares para os problemas de empacotamento. Porém, como no problema *Bin Packing* temos a possibilidade de resolvê-lo utilizando mais de um recipiente, o passo inicial da heurística proposta para este problema consiste na seleção de um recipiente, que é feita pela heurística *First-Fit* (FF).

Para cada item a ser empacotado, a FF trabalha sobre a sequência de recipientes abertos no empacotamento, buscando empacotar o item i no primeiro recipiente que encontrar disponível. Caso não hajam recipientes disponíveis, um novo recipiente é aberto e o item i é ali empacotado. Esse processo de empacotamento se repete para cada um dos itens da lista inicial, sendo a base para o funcionamento do algoritmo para o *Bin Packing*.

Por fim, note que, para este problema, apenas adicionamos uma heurística de seleção de recipientes como passo inicial da heurística de empacotamento completa, e, uma vez selecionado, tal heurística utilizará as mesmas heurísticas elementares anteriormente descritas, similar à proposta para os problemas *Strip Packing* e *Knapsack* apresentada, porém seguindo suas próprias restrições.

5.3.2 Heurísticas adaptadas

Uma característica em comum na resolução dos problemas *Knapsack* e *Bin Packing* se refere às dimensões fixas dos recipientes utilizados. Com isso, durante um empacotamento, há a possibilidade de itens não serem empacotados, quando tomados recipientes específicos. No entanto, muitas vezes isso se deve à disposição dos itens dentro do respectivo recipiente, de forma que o reempacotamento dos mesmos itens, mas de uma forma diferente, poderá permitir que mais itens sejam agregados à sua solução parcial. Com base nisso, propomos uma adaptação das heurísticas básicas desenvolvidas para os problemas *Knapsack* e *Bin Packing*, integrando nestas a heurística desenvolvida para o problema *Strip Packing*.

Com a integração das heurísticas básicas, o algoritmo desenvolvido executará da se-

guinte forma: tomado um recipiente específico, os itens nele contidos são retirados e, em seguida, reempacotados com o uso do algoritmo para o problema *Strip Packing*. Essa execução simples possibilita tornar o respectivo recipiente mais compacto, podendo permitir, dessa forma, que novos itens sejam empacotados. Assim, a heurística adaptada para os problemas *Bin Packing* e *Knapsack* utilizará as suas heurísticas básicas juntamente com a heurística básica proposta para o problema *Strip Packing*, que atuará como uma espécie de compactador da solução parcial contida no recipiente, no qual não foi possível realizar o empacotamento.

O algoritmo completo é o seguinte: dados a lista I de itens a serem empacotados, o algoritmo para o problema *Strip Packing* e o problema de empacotamento a ser resolvido, as heurísticas básicas são executadas até o preenchimento do recipiente $r(j)$, de forma que o item i não caiba nele. Nesse momento, os itens de $r(j)$ são reempacotados pelo o algoritmo do problema *Strip Packing*. Terminada esta otimização, uma nova tentativa de empacotamento é feita com o item i em $r(j)$: caso seja possível, i é posicionado em seu lugar; caso contrário, o processo de escolha/compactação é feito até que não restem mais recipientes abertos a se analisar; caso nenhum recipiente comporte i, um novo recipiente é aberto, quando o problema a ser resolvido é o *Bin Packing*, ou o item i é descartado, quando o problema a ser resolvido é o *Knapsack*. A proposta de integração dos algoritmos para os problemas de empacotamento, que é a base da heurística adaptada para os problemas *Bin Packing* e *Knapsack*, pode ser vista no Algoritmo 5. Note que os recipientes que foram compactados, mas ainda não receberam mais itens, não passarão por um novo processo de compactação, até que um novo item faça parte de sua solução parcial.

Um ponto a se notar na proposta de integração dos algoritmos dos problemas de empacotamento é que o tempo gasto de execução das compactações realizadas pelo algoritmo do problema *Strip Packing* pode interferir negativamente na solução final, caso tais compactações não contribuam para a melhoria do agrupamento dos recipientes. Assim, nesses casos, o tempo utilizado na tentativa de compactar um recipiente faz com que uma menor quantidade de novos indivíduos seja gerada pelo AG, causando uma menor exploração do espaço de soluções. Com isso, a utilização da compactação na heurística adaptada para os problemas *Bin Packing* e *Knapsack* foi condicionada ao comportamento do melhor indivíduo da população do AG, da seguinte forma: para todo indivíduo, é gerado o empacotamento pela heurística básica e armazena-se o *fitness* da solução. Para o melhor indivíduo da população, armazenam-se os valores de *fitness* quando utilizadas a heurística básica e a heurística adaptada. Porém, para os novos indivíduos, a heurística adaptada é executada somente quando algum destes gerar, com o uso da heurística básica, um *fitness* que seja melhor do que o *fitness* do melhor indivíduo da população, considerando seu valor calculado pela heurística básica também. Neste caso, a heurística adaptada é executada

Algoritmo 5: HEURISTICAINTEGRADA(I, *compactador*, *problema*)

Entrada: I: lista de itens a ser empacotado;
compactador: algoritmo para o problema *Strip Packing*;
problema: problema de empacotamento a ser resolvido.

Saída: O conjunto R de recipientes com a solução do empacotamento.

```
Início
    Inicializa(R);
    rec_compactado[r] ← falso;
    Enquanto I ≠ ∅ faça
        empacotado ← falso;
        Para r ← 0 até conta(R) & ¬empacotado faça
            empacotado ← TentaEmpacotar(I, i, R[r]);
            Se empacotado então
                I ← I \ i;
                rec_compactado[r] ← falso;
            senão se ¬rec_compactado[r] então
                Compacta(R[r], compactador);
                empacotado ← TentaEmpacotar(I, i, R[r]);
                Se empacotado então
                    I ← I \ i;
                    rec_compactado[r] ← falso;
                senão
                    rec_compactado[r] ← verdadeiro;
        Se ¬empacotado então
            Se problema = KNAPSACK então
                I ← I\ Retira(I);
            senão
                r ← r + 1;
                TentaEmpacotar(I, i, R[r]);
                I ← I \ i;
                rec_compactado[r] ← falso;
    Retorna R
Fim
```

somente para este indivíduo, que poderá se tornar o melhor da população, caso o *fitness* gerado seja melhor ou igual ao *fitness* gerado pela heurística adaptada do melhor indivíduo encontrado até o momento. É importante destacar que a melhor solução armazenada do

melhor indivíduo da população, que corresponde àquela gerada pela heurística adaptada, sempre será, no mínimo, tão boa quanto a sua solução gerada com o uso da heurística básica, a fim de evitar que uma compactação diminua a qualidade de sua solução final.

Capítulo 6

Configurações e resultados

Este capítulo apresenta os resultados dos algoritmos implementados no trabalho, tanto para os geradores de NFPs quanto para as heurísticas propostas para os problemas empacotamento tratados. Todas as configurações paramétricas dos algoritmos e das instâncias de teste também foram descritas.

6.1 Introdução

Até aqui, detalhamos toda a estrutura elementar das novas heurísticas propostas para os problemas de empacotamento considerados no trabalho, além de termos feito uma descrição completa dos algoritmos utilizados para obtenção das estruturas geométricas de interesse (NFPs).

Deste ponto em diante, descrevemos as configurações paramétricas experimentais, os resultados e as análises feitas para cada um dos algoritmos implementados, indicando, também, os demais componentes utilizados para a obtenção dos resultados. Para isso, iniciamos com a descrição de toda a configuração de ambiente de execução, detalhando os componentes e bases de dados utilizadas para a realização dos testes por cada algoritmo implementado. Em seguida, descrevemos os resultados obtidos pelos algoritmos, diante das configurações específicas apresentadas: primeiramente, descrevemos as configurações específicas e os resultados e análises para os algoritmos geradores de NFPs; ao final, apresentamos estas mesmas informações para as heurísticas propostas para a resolução dos problemas de empacotamento *Strip Packing*, *Knapsack* e *Bin Packing*. Apresentamos, também, uma análise do desempenho do AG na obtenção da solução para o problema *Strip Packing*.

6.2 Configurações básicas

A seguir, apresentamos as configurações básicas utilizadas por todos os algoritmos implementados neste trabalho.

6.2.1 Implementação e ambiente de execução

Todos os testes foram executados em um processador Xeon 2.4GHz CPU, 8GB RAM.

Como base para a execução, utilizamos o sistema operacional Ubuntu GNU/Linux, com a versão do *kernel* 2.6.32-33-generic.

Os algoritmos foram implementados com o uso da linguagem de programação C++, utilizando o padrão da linguagem definido em 2003, compilados com o uso do GCC 4.6, com a opção -O3.

6.2.2 Componente adicional

O tratamento das operações geométricas dos algoritmos deste trabalho foi feito com o uso da biblioteca geométrica chamada CGAL [10].

A CGAL (do inglês *Computational Geometry Algorithms Library*) é uma biblioteca desenvolvida em C++ que contém elementos e algoritmos aplicáveis da área de geometria computacional. Por se tratar de um *software* livre para uso acadêmico, é bastante utilizada em projetos de cunho geométrico. Além disso, há outras características importantes desta biblioteca que justificam seu uso, tais como eficiência, generalidade, robustez, facilidade de uso etc.. Mais informações sobre as características desta biblioteca estão disponíveis em seu próprio site, no endereço `http://www.cgal.org`.

Utilizamos a versão 4.0 da biblioteca CGAL, lançada em março de 2012. Destacamos que não escolhemos versões mais recentes desta biblioteca devido tais lançamentos não afetarem os componentes, estruturas de dados e algoritmos geométricos utilizados em nosso trabalho.

6.2.3 Experimentais

Esta seção descreve as configurações gerais para todos os experimentos realizados. Dentre elas, apresentamos as instâncias utilizadas e suas configurações para os experimentos.

Instâncias dos problemas

Utilizamos duas bases de dados conhecidas da literatura, sendo bases de testes para vários trabalhos que tratam problemas de corte e empacotamento.

A primeira base corresponde a um conjunto de instâncias com itens irregulares obtidos do repositório do ESICUP [26]. ESICUP (do inglês "*EURO Special Interest Group on Cutting and Packing*") é um grupo fundado em 1988 que reúne assuntos referentes a problemas de corte e empacotamento da literatura, apresentando tanto instâncias de problemas (regulares, irregulares, 2D, 3D, geradores de instâncias etc.) quanto descrições de abordagens para obtenção de soluções. Todas as informações estão disponíveis no site do próprio grupo, no endereço `http://paginas.fe.up.pt/~esicup/tiki-index.php`.

Como o nosso trabalho foca na versão irregular dos problemas de empacotamento, utilizamos a base de dados do ESICUP que contém itens irregulares, composta por polígonos simples sem buraco. Dentre estas, há bases compostas por apenas polígonos convexos (fu), bases de quebra-cabeça conhecidas como *jigsaw puzzles* (dighe1 e dighe2) e bases irregulares compostas por polígonos simples convexos e não convexos (demais instâncias). Vale ressaltar que a base "shapes2", utilizada em nossos testes, corresponde à mesma base

Tabela 6.1: Instâncias compostas por itens irregulares (ESICUP [26]).

Nome da instância	Total de itens	Itens distintos	Média de vértices
albano	24	8	6.83
blaz2	20	4	7.50
dagli	30	10	6.20
dighe1	16	16	3.88
dighe2	10	10	4.70
fu	12	11	3.58
han	23	20	7.39
jakobs1	25	22	6.00
jakobs2	25	22	5.36
mao	20	9	8.70
marques	24	8	6.75
poly1a	15	15	4.60
poly2b	30	30	4.93
poly3b	45	45	4.93
poly4b	60	60	4.93
poly5b	75	75	4.84
shapes0	43	4	9.09
shapes1	43	4	9.09
shapes2	28	7	6.29
shirts	99	8	6.05
swim	48	10	20.00
trousers	64	17	6.06

"blaz1" disponibilizada pelo ESICUP, mas que não está incluída em nosso conjunto de testes devido a equivalência entre estas. As informações de cada uma das instâncias utilizadas da base ESICUP podem ser vistas na Tabela 6.1. A referida tabela, para cada linha, nos apresenta as seguintes informações: o nome da instância; o número total de polígonos; o número de polígonos distintos da instância; e o número médio de vértices da respectiva instância.

A segunda base de dados para teste dos algoritmos implementados é a descrita no trabalho de Terashima-Marín *et al.* [41], denominada TERASHIMA1. A base TERASHIMA1 corresponde a um conjunto de instâncias composto por polígonos convexos gerados de forma aleatória, possuindo de 3 a 8 arestas cada um. A criação das instâncias foi dividida em 18 classes distintas, de forma que, para cada uma destas classes, foram geradas 30 instâncias, resultando, assim, em um total de 540. Além das suas próprias instâncias, Terashima-Marín *et al.* adicionaram a instância "fu" (disponível no repositório do ESICUP), que sofreu um ajuste de escala de fator 10. Os detalhes destas instâncias

Tabela 6.2: Instâncias compostas por itens irregulares convexos (Terashima *et al.* [41]).

Tipo da classe	Número de instâncias	Itens por instância	Total de itens	Média de vértices
fu	1	12	12	3.58
A	30	30	900	3.83
B	30	30	900	4.00
C	30	36	1080	3.79
D	30	60	1800	3.54
E	30	60	1800	3.41
F	30	30	900	3.71
G	30	36	1080	4.00
H	30	36	1080	4.00
I	30	60	1800	4.00
J	30	60	1800	3.93
K	30	54	1620	3.69
L	30	30	900	3.59
M	30	40	1200	3.68
N	30	60	1800	3.75
O	30	28	840	3.79
P	30	56	1680	3.41
Q	30	60	1800	4.00
R	30	54	1620	3.71
Total	541	832	24612	

são apresentados na Tabela 6.2; em cada linha desta tabela, temos a seguinte informação: nome do tipo da classe; número de instâncias que constituem a classe; número de itens em cada uma das instâncias; número total de itens na classe; e número médio de vértices de todos os itens da classe. Por fim, os demais detalhes de criação destas instâncias podem ser encontrados no trabalho de Terashima-Marín *et al.* [41].

6.3 Configurações específicas e resultados

Esta seção apresenta as configurações paramétricas específicas adotadas, tanto para as instâncias das bases de teste utilizadas quanto para as heurísticas e algoritmos implementados neste trabalho. Dadas tais configurações, os resultados computacionais são apresentados e discutidos.

6.3.1 Geradores de NFPs

Configurações dos testes dos geradores de NFPs

O experimento realizado para analisar o desempenho dos geradores de NFPs foi executado uma única vez, de modo que foram gerados os NFPs entre todos os pares de itens (inclusive de um item com ele próprio), para cada uma das instâncias da base ESICUP –descrita na seção 6.2.3–, considerando apenas os polígonos distintos de cada uma. Apesar disso, note que quando há repetição de instâncias (i.e., instâncias de nomes diferentes que possuem o mesmo conjunto de polígonos, como blaz1 e shapes2, e shapes0 e shapes1), apenas uma destas foi utilizada na execução dos experimentos. O último ponto está ligado à rotação dos polígonos distintos de cada uma das instâncias, limitada ao conjunto formado pelos ângulos $0\,^\circ$, $90\,^\circ$, $180\,^\circ$ e $270\,^\circ$, resultando em quatro polígonos distintos.

Para os três geradores, fizemos uso dos três algoritmos apresentados no capítulo 4: o primeiro, denominado Gen1, é a implementação proposta no trabalho de Burke *et al.* [7], baseada no método orbital; o segundo, denominado Gen2, é uma implementação feita com o uso da biblioteca geométrica CGAL [10], baseada na SM via decomposição; o último, denominado Gen3, é a implementação descrita por Bennell e Song [5], baseada na SM via adição de borda.

Resultados e discussão

Dadas as configurações experimentais para os geradores de NFPs, apresentamos os resultados obtidos por cada algoritmo implementado.

Tabela 6.3: Tempo gasto pelos geradores de NFPs – instância original.

Nome	Dstc	M. vér.	NFPs	Gen1(seg)	Gen2(seg)	Gen3(seg)
albano	8	7.25	1024	6.59	3.04	**1.29**
blaz2	4	7.50	256	1.40	0.40	**0.22**
dagli	10	6.20	1600	8.21	2.19	**1.43**
dighe1	16	3.88	4096	6.68	1.38	**0.51**
dighe2	10	4.70	1600	3.44	0.83	**0.29**
fu	11	3.55	1936	2.95	0.42	**0.21**
han	20	7.35	6400	64.57	16.05	**15.08**
jakobs1	22	5.45	7744	30.39	**5.07**	11.62
jakobs2	22	5.41	7744	35.62	9.45	**8.09**
mao	9	9.22	1296	18.53	6.58	**3.35**
marques	8	7.13	1024	6.31	2.83	**1.54**
poly1a	15	4.60	3600	7.88	2.82	**0.81**
poly2b	30	4.93	14400	36.23	11.88	**3.95**
poly3b	45	4.93	32400	79.79	26.67	**8.46**
poly4b	60	4.93	57600	140.33	45.04	**14.42**
poly5b	75	4.84	90000	212.35	68.73	**21.63**
shapes1	4	8.75	256	4.30	**0.91**	3.52
shapes2	7	6.29	784	3.12	1.05	**0.42**
shirts	8	6.63	1024	5.30	1.36	**0.99**
swim	10	21.90	1600	197.27	71.59	**19.57**
trousers	17	5.06	4624	12.43	2.68	**1.07**
Tempo total				**883.69**	**280.97**	**118.47**

A Tabela 6.3 expressa, para cada algoritmo, o tempo gasto (em segundos) para a geração de todos NFPs, para cada uma das instâncias. Cada linha desta tabela apresenta a seguinte informação: o nome da instância (Nome); o número de polígonos distintos da instância (Dstc); o número médio de vértices dos itens distintos (M. vér.); o total de NFPs do conjunto de itens distintos, dadas as 4 rotações utilizadas (NFPs); o tempo gasto para gerar todos os NFPs dos itens distintos para o método orbital (Gen1); o tempo gasto para gerar todos os NFPs dos itens distintos para a SM via decomposição (Gen2); o tempo gasto para gerar todos os NFPs dos itens distintos para a SM via adição de borda (Gen3).

Pelos resultados da Tabela 6.3, podemos observar uma grande diferença nos tempos de execução dos geradores. Na maioria dos casos, Gen3 apresenta os melhores resultados, chegando a ser, em média, acima de 7 vezes mais rápido que Gen1. No entanto, destacamos que Gen3 não trabalha com polígonos com buracos, o que não descarta a possibilidade de utilização de Gen1 em problemas com polígonos desse tipo. Para Gen2, há dois casos em que o tempo de geração é o menor em comparação aos outros dois geradores, mas

Tabela 6.4: Tempo gasto pelos geradores de NFPs – instância modificada.

Nome	Dstc	M. vér.	NFPs	Gen1(seg)	Gen2(seg)	Gen3(seg)
albano	8	7.63	1024	7.58	4.02	**1.71**
blaz2	4	8.00	256	1.61	0.70	**0.35**
dagli	10	6.90	1600	11.99	4.79	**1.96**
dighe1	16	4.81	4096	11.20	6.19	**1.81**
dighe2	10	5.50	1600	4.96	2.60	**0.91**
fu	11	4.55	1936	4.38	2.53	**0.85**
han	20	7.80	6400	74.44	23.08	**17.93**
jakobs1	22	6.14	7744	37.99	**13.15**	16.51
jakobs2	22	6.00	7744	42.90	17.41	**11.55**
mao	9	9.56	1296	20.09	8.45	**3.90**
marques	8	7.50	1024	7.29	3.77	**2.01**
poly1a	15	5.27	3600	10.58	6.40	**1.77**
poly2b	30	5.57	14400	48.71	26.57	**7.88**
poly3b	45	5.56	32400	104.83	57.69	**17.12**
poly4b	60	5.57	57600	186.51	100.55	**30.67**
poly5b	75	5.47	90000	288.11	155.12	**46.52**
shapes1	4	9.00	256	4.40	**1.07**	3.78
shapes2	7	6.86	784	3.69	1.90	**0.73**
shirts	8	7.13	1024	6.04	2.23	**1.32**
swim	10	22.00	1600	201.52	75.40	**21.14**
trousers	17	5.94	4624	15.80	8.57	**2.54**
Tempo total				**1094.62**	**522.19**	**192.96**

não a ponto de justificar a sua utilização, vista sua característica de não trabalhar com polígonos com buraco, nem gerar as arestas coincidentes e pontos interiores.

Apesar dos resultados obtidos por Gen3 geralmente serem os que apresentam o menor tempo de execução, há um ponto que devemos destacar. Pelos resultados obtidos, é evidente que quanto mais vértices um polígono possui, mais custoso será a geração do NFP correspondente. Não obstante, a convexidade de um polígono também é determinante nesse tempo gasto, como pode ser observado ao se comparar as instâncias dighe2 e dagli, no qual esta possui uma quantidade maior de polígonos não convexos.

Visando analisar o impacto desta não convexidade no processo de criação de NFPs pelos algoritmos, aplicamos uma alteração nos itens destas mesmas instâncias: para cada item convexo, adicione um vértice reflexo de forma que o item se torne um polígono simples não convexo; feito isso, os NFPs entre todos esses pares foram gerados de forma similar à execução anterior. Os resultados para esta alteração são apresentados na Tabela 6.4.

Tabela 6.5: Porcentagem de aumento de tempo de geração de NFPs após alteração dos itens.

Nome	Gen1(%)	Gen2(%)	Gen3(%)
albano	15.02	32.24	32.56
blaz2	15.00	75.00	59.09
dagli	46.04	118.72	37.06
dighe1	67.66	348.55	254.90
dighe2	44.19	213.25	213.79
fu	48.47	502.38	304.76
han	15.29	43.80	18.90
jakobs1	25.01	159.37	42.08
jakobs2	20.44	84.23	42.77
mao	8.42	28.42	16.42
marques	15.53	33.22	30.52
poly1a	34.26	126.95	118.52
poly2b	34.45	123.65	99.49
poly3b	31.38	116.31	102.36
poly4b	32.91	123.25	112.69
poly5b	35.68	125.69	115.07
shapes1	2.33	17.58	7.39
shapes2	18.27	80.95	73.81
shirts	13.96	63.97	33.33
swim	2.15	5.32	8.02
trousers	27.11	219.78	137.38

Comparando os resultados das Tabelas 6.3 e 6.4, nota-se que a não convexidade dos itens interfere muito no tempo de execução dos geradores, pois a adição de apenas um vértice deixando os polígonos não convexos fez com que o tempo de execução sofresse bastante alteração, principalmente nas técnicas baseadas na SM. Para uma melhor comparação, a Tabela 6.5 apresenta a porcentagem da variação de tempo entre os geradores.

6.3.2 Heurísticas propostas

Configurações dos testes das heurísticas propostas

Considerando os problemas de empacotamento, foram realizados um total de 10 execuções do respectivo algoritmo, para cada instância utilizada em cada problema. Destacamos que, à cada execução de um novo experimento, configuramos uma nova semente para o gerador de números aleatórios do algoritmo.

Em relação à resolução das bases de teste pelas heurísticas (seção 6.2.3), foram tomadas

Tabela 6.6: Configuração paramétrica de ESICUP para os problemas *Strip Packing* e *Knapsack*.

Instância	Altura(h)	Rotação(°)	Tempo(seg)
albano	4900	0;180	1200
blaz2	15	0;180	1200
dagli	60	0;180	1200
dighe1	100	0	600
dighe2	100	0	600
fu	38	0;90;180;270	600
han	58	0;90;180;270	1200
jakobs1	40	0;90;180;270	600
jakobs2	70	0;90;180;270	600
mao	2550	0;90;180;270	1200
marques	104	0;90;180;270	1200
poly1a	40	0;90;180;270	1200
poly2b	40	0;90;180;270	1200
poly3b	40	0;90;180;270	1200
poly4b	40	0;90;180;270	1200
poly5b	40	0;90;180;270	1200
shapes0	40	0	1200
shapes1	40	0;180	1200
shapes2	15	0;180	1200
shirts	40	0;180	1200
swim	5752	0;180	1200
trousers	79	0;180	1200

da seguinte forma: para os problemas *Strip Packing* e *Knapsack*, utilizamos as instâncias da base ESICUP, com as configurações de itens e recipiente dos problemas especificadas na Tabela 6.6. Note que está definida apenas a altura do recipiente a ser utilizado, pois esses são valores padrão para o problema *Strip Packing*. Para o problema *Knapsack*, o comprimento definido será apresentado posteriormente. Ressaltamos que as rotações permitidas para cada instância foram obtidas dos trabalhos mais recentes para o problema *Strip Packing*.

Para o problema *Bin Packing*, utilizamos o conjunto de instâncias irregulares da base TERASHIMA1. Algumas de suas características estão descritas na Tabela 6.7.

É importante destacar que, para algumas instâncias, o tempo de execução foi ligeiramente superior ao descrito nas Tabelas 6.6 e 6.7, pelo fato de encerrarmos a execução do algoritmo somente ao término da construção de uma solução, caso esta ainda esteja sendo gerada no final de uma execução. Além disso, a comparação dos resultados obtidos

Tabela 6.7: Configuração paramétrica de TERASHIMA1 para o problema *Bin Packing*.

Tipo	Rotação(°)	Tempo(seg)	Solução ótima (n°. de recip.)
fu	0	120	*
A	0	120	3
B	0	120	10
C	0	120	6
D	0	120	3
E	0	120	3
F	0	120	2
G	0	120	*
H	0	120	12
I	0	120	3
J	0	120	4
K	0	120	6
L	0	120	3
M	0	120	5
N	0	120	2
O	0	120	7
P	0	120	8
Q	0	120	15
R	0	120	9

* Valor não conhecido.

pelos algoritmos, quando consideramos tempo de execução, deve ser vista com cautela, uma vez que o ambiente de execução destes algoritmos pode ter configurações diferentes.

As heurísticas propostas para os problemas de empacotamento foram ajustadas com os seguintes parâmetros comuns: verificação de sobreposição dos itens baseada na técnica de NFPs, com a geração destas estruturas pelo método de adição de borda para polígonos simples –seção 4.3.2–; parâmetro α equivalente a 5%, com o referido teste baseando-se no aproveitamento do agrupamento dos itens considerando a envoltória convexa destes; e heurística de agrupamento, quando utilizada a Busca Discretizada ou a Busca Aleatória, avaliando o aproveitamento do agrupamento com uso da envoltória convexa dos itens. Em relação a estes tipos de buscas, temos as seguintes configurações: para a Busca Discretizada, foi tomado um intervalo para geração de pontos com tamanho referente à 20% do perímetro do NFP utilizado. Este valor é reduzido à metade quando não se encontra ao menos uma posição de empacotamento que esteja dentro do recipiente. Para a Busca Aleatória, o conjunto de pontos foi formado pelas seguintes alternativas: 20 pontos aleatórios sobre a borda externa do NFP ou 20 pontos aleatórios sobre cada uma de suas

arestas, sendo esta uma escolha de igual probabilidade. Note que, para qualquer tipo de busca, caso não se encontre uma posição viável de empacotamento, o respectivo item é excluído da solução.

Os parâmetros para o AG foram ajustados seguindo um teste de força bruta, para um conjunto fixo de parâmetros; cada configuração foi executada por um tempo de 1200 segundos. Os parâmetros mais promissores dessa configuração paramétrica estão descritos na Tabela 6.8.

Tabela 6.8: Configuração paramétrica do AG.

Parâmetro	Configuração
População	4
Descendentes por indivíduo	1
Busca Estendida	90%
Busca Discretizada	5%
Busca Aleatória	5%
Swap	33%
Inserção	34%
Inverção	33%
Reinicialização da população	300 segundos
Seleção	Torneio

Por fim, para resolver os problemas *Bin Packing* e *Knapsack*, quando se utiliza a heurística adaptada – descrita na seção 5.3.2 –, foi fixado um tempo de 2 e 3 segundos de otimização para o algoritmo do problema *Strip Packing*, respectivamente.

Resultados e discussão

Os resultados obtidos para cada algoritmo implementado estão descritos a seguir. Ao final, uma análise de desempenho do AG também é apresentada.

Strip Packing

Os resultados da heurística básica proposta para o problema *Strip Packing* estão descritos na Tabela 6.9. São apresentados, também, os resultados obtidos da literatura. Cada linha da referida tabela contém a seguinte informação: nome da instância do problema; utilização da melhor solução encontrada (Melhor Util), utilização média gerada (Média Util) e tempo médio gasto (Tempo) pela heurística proposta; e estas mesmas informações para as heurísticas da literatura com os melhores resultados. Para facilitar

Tabela 6.9: Resultados da heurística proposta para o problema *Strip Packing*.

Nome da instância	Heurística Básica Proposta			Literatura		
	Melhor	Média		Melhor	Média	
	Util(%)	Util(%)	Tempo(seg)	Util(%)	Util(%)	Tempo(seg)
albano	84.79	82.93	1200.20	**88.48**[a]	87.38[a]	1203.00
blaz2	**74.68**	73.94	1200.00	*	*	*
dagli	82.20	80.50	1200.20	**88.11**[a]	86.27[a]	1205.00
dighe1	**100.00**	82.09	1119.00	**100.00**[ab]	93.93[c]	600.00
dighe2	**100.00**	94.16	429.20	**100.00**[ab]	100.00[a]	600.00
fu	86.36	86.08	1200.00	**91.94**[a]	90.93[c]	600.00
han	**77.95**	76.16	1200.40	*	*	*
jakobs1	81.67	81.67	1200.50	**89.10**[a]	88.90[c]	600.00
jakobs2	74.23	71.18	1200.30	**83.92**[a]	80.97[a]	602.00
mao	81.18	79.39	1200.90	**85.15**[c]	82.79[f]	1200.00
marques	85.93	84.79	1200.80	**89.73**[a]	88.80[f]	1200.00
poly1a	73.15	70.86	1200.00	**73.20**[d]	*	*
poly2b	71.62	70.38	1201.40	**75.40**[d]	*	*
poly3b	73.68	71.85	1203.80	**74.60**[d]	*	*
poly4b	74.34	72.19	1206.00	**74.70**[d]	*	*
poly5b	73.25	71.93	1208.60	**74.70**[d]	*	*
shapes0	59.55	58.59	1200.60	**68.44**[e]	66.85[a]	1207.00
shapes1	65.41	63.07	1201.60	**75.29**[a]	74.24[a]	1212.00
shapes2	77.60	75.49	1200.20	**84.25**[e]	82.55[a]	1205.00
shirts	84.05	81.97	1209.40	**88.78**[e]	88.12[e]	1200.00
swim	68.90	65.57	1220.50	**75.43**[a]	74.62[e]	1200.00
trousers	87.38	85.89	1202.30	**89.96**[b]	89.29[c]	600.00

[a] Algortimo ELS de Leung *et al.* [29].
[b] Algortimo SAHA de Gomes e Oliveira [22].
[c] Algoritmo 2DNest de Egeblad *et al.* [17].
[d] Algoritmo baseado na Busca Tabu de Burke *et al.* [7].
[e] Algoritmo ILSQN de Imamichi *et al.* [27].
[f] Algoritmo FITS de Umetani *et al.* [42].
* Valor indisponível para as configurações utilizadas.

a análise, destacamos os valores da melhor utilização encontrada (negrito) e da maior utilização média (sublinhado), para cada instância.

O critério para a escolha dos melhores resultados da literatura foi baseado no tempo de execução dos algoritmos, além da obrigatoriedade de se ter a mesma configuração paramétrica para todas as instâncias resolvidas, que estão descritos na Tabela 6.6. Por esse motivo, os resultados obtidos nos trabalhos de Bennell e Song [6], Sato *et al.* [39] e Shalaby e Kashkoush [40] não foram considerados, pelo fato de utilizarem um tempo de

execução consideravelmente maior na otimização das respectivas instâncias. Entretanto, ainda assim é necessária uma comparação cautelosa dos resultados da heurística proposta com os resultados da literatura, ao considerar o tempo utilizado nos experimentos, vistas as diferentes características do ambiente de execução de cada um dos algoritmos.

Com base nos critérios anteriores, os melhores algoritmos que encontramos na literatura para o problema *Strip Packing* são:

- ELS de Leung *et al.* [29];
- SAHA de Gomes e Oliveira [22];
- 2DNest de Egeblad *et al.* [17];
- algoritmo baseado na Busca Tabu de Burke *et al.* [7];
- ILSQN de Imamichi *et al.* [27]; e
- FITS de Umetani *et al.* [42].

O ambiente de execução e o número de execuções de cada um dos experimentos realizados por cada algoritmo são apresentados na Tabela 6.10. Cada linha desta tabela apresenta a seguinte informação: nome do algoritmo; configuração do ambiente de execução para a realização do experimento; e o número de execuções para cada instância utilizada.

Tabela 6.10: Configurações básicas para execução dos algoritmos da literatura.

Algoritmo	Configuração de ambiente	Execuções (n°.)
ELS	Pentium4 2.4GHz CPU, 2GB de RAM.	10
SAHA	Pentium4 2.4GHz CPU, 512MB de RAM.	20
2DNest	Pentium4 3GHz CPU, sem detalhes de RAM.	20
Busca Tabu	Pentium4 2GHz CPU, 256MB de RAM.	10
ILSQN	Intel Xeon 2.8GHz CPU, 1GB de RAM.	10
FITS	Intel Xeon 2.8GHz CPU, 2GB de RAM.	10

Note que cada resultado da literatura apresentado na Tabela 6.9 está vinculado ao algoritmo gerador, sendo que a média de tempo de execução corresponde ao algoritmo de melhor média. Vale ressaltar que no trabalho de Burke *et al.*, são propostas duas variações de algoritmo (Busca Tabu e *Hill Climbing*) que resolvem o problema sob duas formas distintas de ordenação dos itens. Logo, consideramos apenas aquela que obteve os melhores resultados (maior quantidade de resultados com maior utilização), referindo-se à heurística Busca Tabu com o conjunto de itens ordenado por área.

A Tabela 6.9 indica uma grande variação de alguns resultados obtidos pela heurística proposta em comparação com os resultados da literatura. Um possível motivo dessa variação seria a forma que a biblioteca geométrica produz algumas estruturas geométricas utilizadas para a criação das soluções para o problema de empacotamento. Como visto, a HA – seção 5.2.1 – trabalha agrupando os itens de uma lista, utilizando os NFPs dos itens para isso. No entanto, pelo motivo de não gerarmos o NFP resultante à cada novo item empacotado, mas utilizarmos a união dos NFPs dos itens individuais correspondentes, a operação de união dos polígonos feita pela biblioteca geométrica acaba não preservando as arestas coincidentes e pontos interiores do NFP resultante, o que acaba sendo um problema para a heurística de agrupamento de itens, por limitá-la na busca de melhores posições de empacotamento. O mesmo problema pode ser inferido do trabalho de Oliveira *et al.* [36], que desconsidera algumas estruturas geométricas durante o empacotamento, fazendo com que muitos dos resultados obtidos estejam em uma faixa de valores próximos aos da heurística proposta. Já no trabalho de Bennell e Song [6], que é uma evolução do trabalho de Oliveira *et al.* [36], há considerável diferença nos resultados. No entanto, Bennell e Song propuseram alterações na heurística original, sendo uma delas a forma de se trabalhar com a estrutura geométrica da solução parcial para empacotamento dos itens. Por fim, o trabalho de Sato *et al.* [39] também utiliza-se de um método construtivo, destacando que em todas as operações geométricas de união de conjunto são preservadas as arestas coincidentes e os pontos interiores, a fim de permitir um melhor agrupamento dos itens.

Apesar da variação apresentada pelos resultados da Tabela 6.9, alguns valores obtidos atualizam os resultados da literatura com as configurações paramétricas adotadas, como é o caso de blaz2, han, poly1a, poly2b, poly3b, poly4b e poly5b. Destacamos também que, para as instâncias dighe1 e dighe2 (*jigsaw puzzles*), foi encontrada a solução ótima (100% de aproveitamento).

A Figura 6.1 apresenta o melhor resultado encontrado para cada uma das instâncias resolvidas do problema *Strip Packing*.

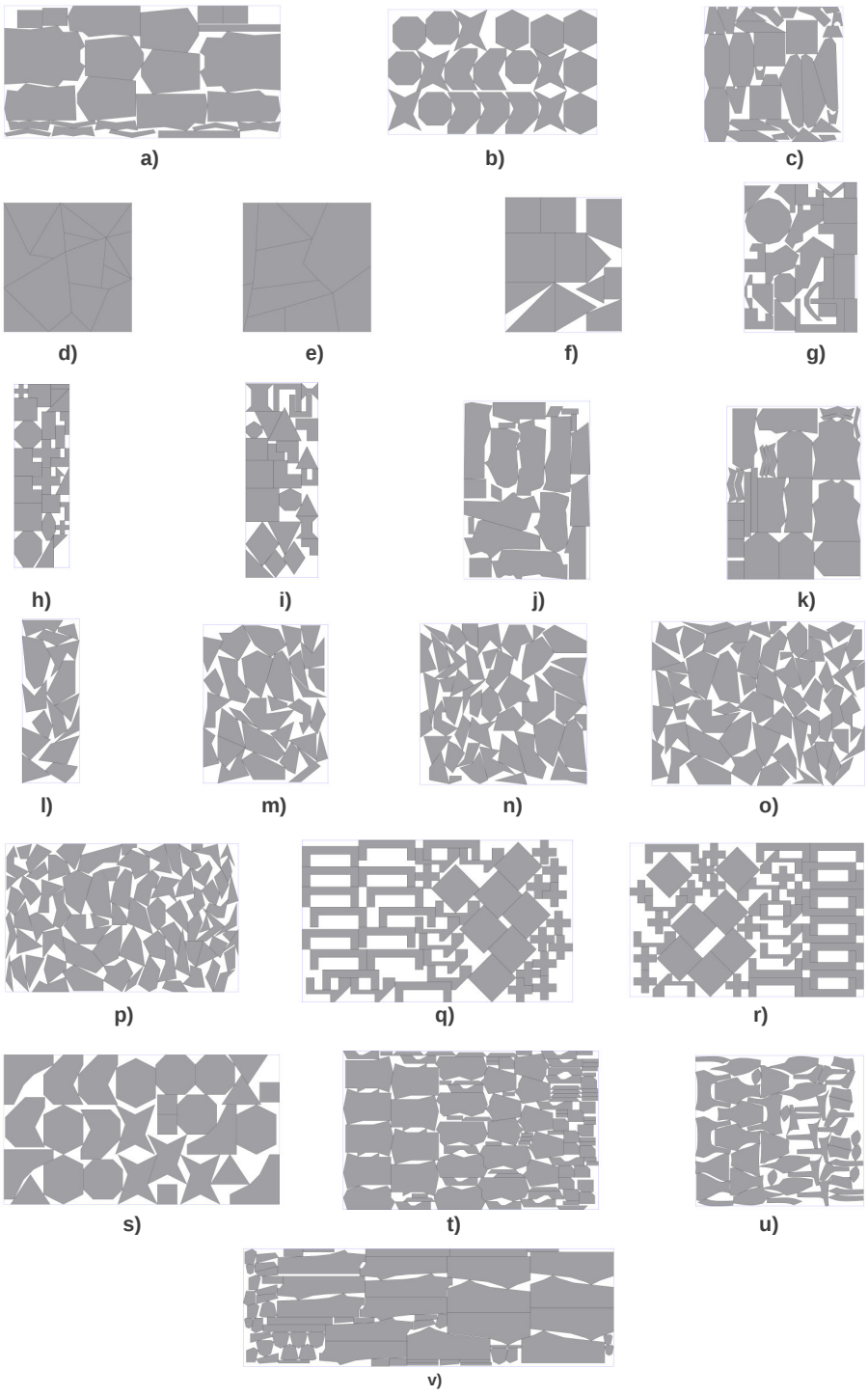


Figura 6.1: Resultados para o problema *Strip Packing* para as instâncias: a) albano; b) blaz2; c) dagli; d)dighe1; e) dighe2; f) fu; g) han; h) jakobs1; i) jakobs2; j) mao; k) marques; l) poly1a; m) poly2b; n) poly3b; o) poly4b; p) poly5b; q) shapes0; r) shapes1; s) shapes2; t) shirts; u) swim; v) trousers.

Knapsack

Os resultados das heurísticas propostas para o problema *Knapsack* estão descritos na Tabela 6.11. A fim de verificar o desempenho de ambas heurísticas (básica e adaptada), apresentamos também os melhores resultados encontrados da literatura. As informações apresentadas nesta tabela, para cada linha, são: nome da instância; largura do recipiente; ocupação máxima da instância com o empacotamento de todos os itens; utilização da melhor solução encontrada (Melhor Util), utilização média gerada (Média Util), quantidade média de itens empacotados (Itens) e tempo médio gasto (Tempo) da heurística básica proposta; e estas mesmas informações para a heurística adaptada proposta e para a heurística da literatura com o melhor resultado encontrado (exceto a utilização da melhor solução da literatura, por não serem encontrados tais resultados). Para facilitar a análise, destacamos os valores da melhor utilização encontrada (negrito) e da maior utilização média (sublinhado), para cada instância.

Tratando-se dos resultados da literatura para instâncias da base ESICUP, encontramos o trabalho de Del Valle *et al.* [14], que considera a maior parte das instâncias resolvidas. Para as instâncias não presentes no referido trabalho, foi utilizado um comprimento de recipiente que representasse o melhor valor encontrado na literatura para o problema *Strip Packing*, considerando apenas resultados com configurações paramétricas semelhantes às descritas na seção 6.3.2. Para as instâncias poly1a, poly2b, poly3b, poly4b e poly5b, foi utilizado um comprimento do recipiente com base no trabalho de Burke *et al.* [7]; para as instâncias blaz2 e han, o respectivo valor refere-se ao melhor resultado da Tabela 6.9 para estas instâncias. Vale ressaltar que não utilizamos os valores obtidos no trabalho de Mundim e Queiroz [34], pois este apresenta, para algumas instâncias, valores de comprimento de recipiente diferente do utilizado por Del Valle *et al.* [14]. Por fim, Del Valle *et al.* utilizam como configuração de ambiente: Core2 Quad 2.4GHz CPU, 4GB de RAM.

Um ponto a se notar, na comparação entre os resultados das heurísticas propostas com o trabalho de Del Valle *et al.* [14], é a padronização de alguns parâmetros que adotamos para a execução do algoritmo, a fim de facilitar comparações com os resultados de futuras técnicas para resolução do problema. Dentre elas, fixamos o conjunto de rotações permitidas para cada instância e o tempo máximo de execução do algoritmo. Tais parâmetros correspondem ao mesmo conjunto utilizado quando resolvemos o problema *Strip Packing*, e estão especificados na Tabela 6.6. Novamente, a comparação dos resultados obtidos com os resultados da literatura, quando levamos em consideração o tempo utilizado, deve ser feita com cautela, devido as diferentes configurações do ambiente de execução.

Comparando os resultados obtidos pelas heurísticas propostas, é visto que a versão básica da heurística obteve melhores resultados em vários aspectos. Primeiramente, apresentou uma maior quantidade de resultados com aproveitamento médio superior. Outro

Tabela 6.11: Resultados das heurísticas propostas para o problema *Knapsack*.

Nome da instância	Larg. do recipient.	Máx. oc.(%)	Heurística Proposta – Básica				Heurística Proposta – Adaptada				Literatura		
			Melhor	Média			Melhor	Média			Média		
			Util(%)	Util(%)	Itens	Tempo(seg)	Util(%)	Util(%)	Itens	Tempo(seg)	Util(%)	Itens	Tempo(seg)
albano	10122.63	86.06	**84.81**	83.84	21.8	1200.5	84.27	83.48	21.4	1203.2	80.38	23	975.99
blaz2	25.20	74.74	**71.83**	71.17	18.8	1200.5	71.83	70.95	18.7	1201.0	*	*	*
dagli	65.60	77.31	**77.31**	77.31	30.0	26.2	**77.31**	77.31	30.0	109.4	75.86	29	1132.56
dighe1	138.13	72.40	**72.40**	72.40	16.0	31.6	**72.40**	72.40	16.0	18.0	72.40	16	10.80
dighe2	134.05	74.60	**74.60**	74.60	10.0	1.7	**74.60**	74.60	10.0	7.8	74.60	10	0.22
fu	34.00	83.82	**83.82**	83.82	12.0	13.2	**83.82**	83.82	12.0	69.7	83.82	12	22.05
han	43.57	77.95	**77.95**	76.71	22.0	1162.7	77.16	76.37	21.7	1200.6	*	*	*
jakobs1	13.00	75.38	**75.38**	75.38	25.0	13.6	**75.38**	75.38	25.0	10.4	75.38	25	67.79
jakobs2	28.20	68.44	**68.44**	68.44	25.0	38.8	**68.44**	68.44	25.0	32.9	68.44	25	619.51
mao	2058.60	71.60	**71.60**	71.60	20.0	10.0	**71.60**	71.60	20.0	31.1	71.60	20	223.33
marques	83.60	82.74	**82.74**	82.74	24.0	58.2	**82.74**	82.74	24.0	63.2	82.74	24	275.27
poly1a	13.90	73.73	71.75	70.15	13.8	1200.0	**73.74**	70.33	13.9	1110.0	*	*	*
poly2b	29.99	75.40	72.65	71.49	27.0	1201.7	**73.88**	72.01	27.8	1201.6	*	*	*
poly3b	40.72	74.90	**73.12**	72.55	42.3	1202.7	72.69	72.22	42.0	1202.8	*	*	*
poly4b	51.70	74.80	**74.34**	73.02	57.1	1206.6	73.21	73.01	56.8	1207.2	*	*	*
poly5b	57.71	79.53	**74.51**	74.09	65.9	1214.3	74.40	73.77	65.5	1214.8	*	*	*
shapes0	63.00	63.33	**61.75**	60.33	40.5	1200.4	61.11	59.71	40.0	1201.6	60.16	41	1603.78
shapes1	59.00	67.63	**65.59**	63.90	39.5	1202.0	**65.59**	63.59	39.6	1203.1	64.24	41	4094.16
shapes2	27.30	79.12	76.43	75.87	26.7	1200.2	**77.66**	76.12	26.8	1201.0	72.89	26	1066.02
shirts	63.13	85.54	**84.71**	83.60	92.1	1207.5	83.89	83.23	89.1	1220.1	77.02	96	14525.62
swim	6568.00	67.35	**67.35**	66.90	47.3	793.1	**67.35**	67.08	47.5	820.9	64.27	46	40869.64
trousers	245.75	88.63	87.45	86.89	59.1	1202.0	**88.30**	87.14	59.3	1206.4	78.66	62	5844.81

* Valor não disponível.

ponto importante foi o tempo gasto, que foi menor na maioria dos casos, com destaque para as instâncias em que se obteve a solução ótima. Diante disso, podemos destacar algumas características da heurística adaptada: o período de tempo gasto na compactação de um recipiente pode interferir negativamente no resultado final quando a otimização feita não resulta em melhorias, pelo fato do tempo despendido não ser aplicado pelo AG para gerar novos indivíduos na população. Isso indica que, para se obter resultados similares aos resultados da heurística básica, nesse caso, seria necessário um período de tempo maior de otimização, como pode ser observado nos resultados obtidos para as instâncias mais simples, nas quais foram encontradas a solução ótima, porém em um período de tempo maior.

Apesar destas características observadas, há casos em que a aplicação da compactação é interessante. Por exemplo, em alguns experimentos em que as instâncias resolvidas são mais complexas e possuem uma quantidade maior de itens distintos (como a swim, trousers etc.), a compactação realizada na solução que está sendo construída pela heurística adaptada tende a gerar melhores soluções.

Na comparação dos resultados obtidos por cada uma das heurísticas propostas com os resultados da literatura, destacamos que obtivemos apenas dois resultados inferiores, quando comparado o aproveitamento médio encontrado pela heurística adaptada, e apenas um resultado inferior, quando comparado o aproveitamento médio encontrado pela heurística básica. No entanto, note que, nesses casos, o tempo médio utilizado por ambas as heurísticas propostas foi menor, para os referidos resultados. Diante disso, um ponto de destaque foi o resultado da padronização de tempo de otimização. Embora começássemos a adotar um tempo limite para conclusão da otimização, muitos problemas foram resolvidos sem que tal limite fosse atingido, mostrando a eficiência das heurísticas propostas. Além disso, os resultados obtidos de ocupação do recipiente para algumas instâncias foi bastante superior, mesmo considerando o limite de tempo, que foi significantemente menor. Em relação ao período de execução para otimização das instâncias em que foram encontradas as soluções ótimas, alguns resultados do trabalho de Del Valle *et al.* mostram um tempo gasto bastante pequeno em comparação com as outras heurísticas, mas que não chega a ser muito expressivo.

Em relação aos resultados da Tabela 6.11, acreditamos que a ordenação inicial dos itens (ordem não crescente de área) foi importante para obtenção dos resultados, visto que, em alguns problemas, quando comparados com os resultados obtidos de Del Valle *et al.*, a quantidade média de itens foi menor, mesmo gerando uma ocupação maior do recipiente. Outro ponto importante é em relação ao tempo de geração de NFPs, que apresentou um valor insignificante quando comparado ao tempo total de otimização.

O melhor empacotamento encontrado para cada instância do problema *Knapsack* é apresentado na Figura 6.2.

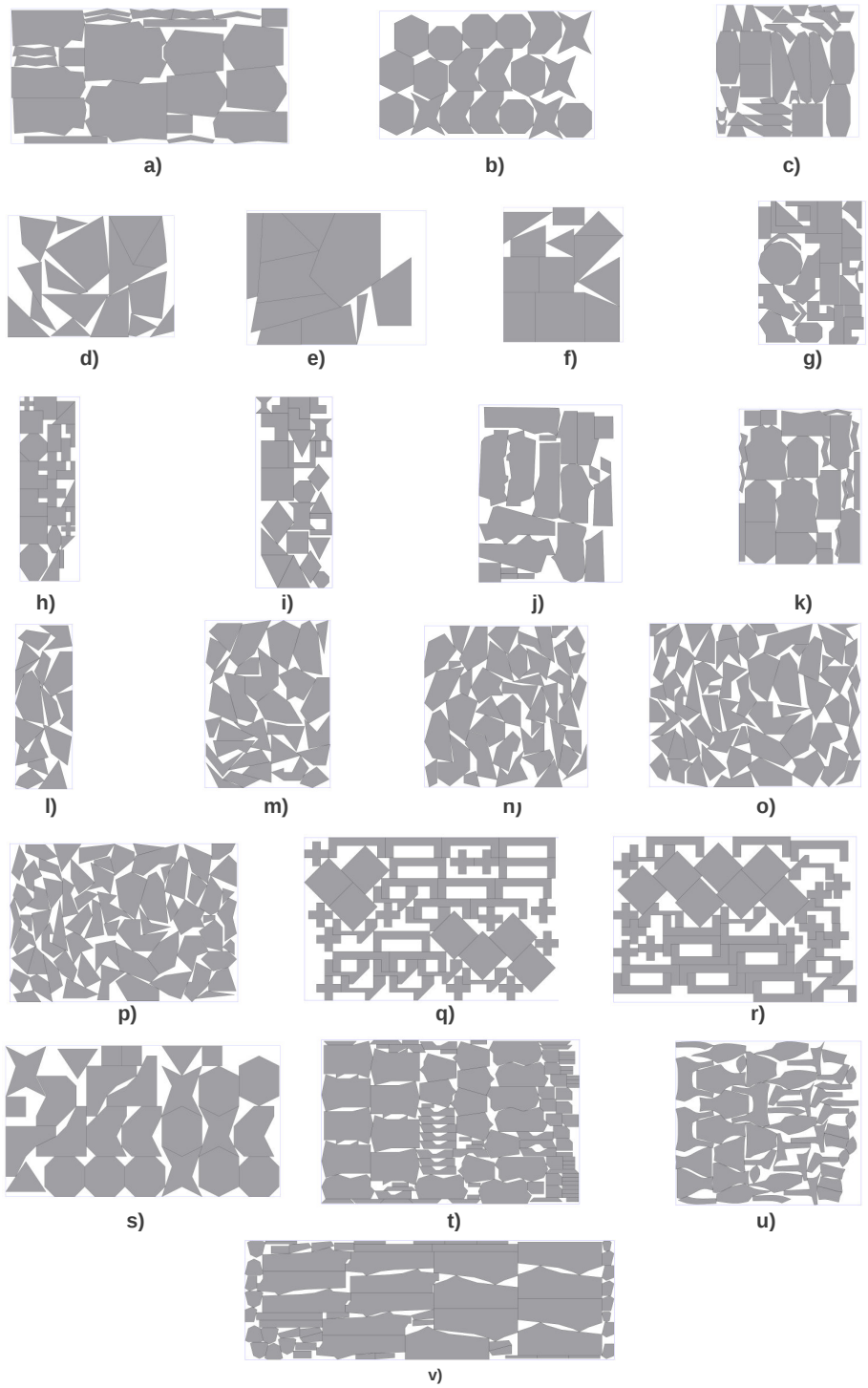


Figura 6.2: Resultados para o problema *Knapsack* para as instâncias: a) albano; b) blaz2; c) dagli; d)dighe1; e) dighe2; f) fu; g) han; h) jakobs1; i) jakobs2; j) mao; k) marques; l) poly1a; m) poly2b; n) poly3b; o) poly4b; p) poly5b; q) shapes0; r) shapes1; s) shapes2; t) shirts; u) swim; v) trousers.

Bin Packing

Considerando o problema *Bin Packing*, apresentamos os resultados obtidos pelas heurísticas propostas na Tabela 6.12. Para analisar o desempenho de ambas as heurísticas (básica e adaptada) sobre as instâncias resolvidas, apresentamos os resultados em detalhes para até quatro recipientes extras em relação à solução ótima (ou em relação à solução gerada que utilizou a menor quantidade de recipientes dentre os experimentos realizados, caso a solução ótima não esteja disponível). Cada linha desta tabela apresenta a seguinte informação: nome da classe (Nome); número de recipientes utilizados na solução ótima (OPT); menor número de recipiente utilizados (Min.), maior número de recipientes utilizados (Máx.) e utilização média de recipientes (Méd.) encontradas, tempo médio das execuções (Tempo), porcentagem de soluções com zero (0), um (1), dois (2), três (3), quatro (4) ou cinco ou mais ($\geq$5) recipientes extras em relação à solução ótima (ou em relação à solução gerada que utilizou a menor quantidade de recipientes dentre os experimentos realizados, se a solução ótima for desconhecida) para os resultados produzidos pela heurística básica proposta; e estas mesmas informações, para a heurística adaptada proposta.

Os resultados da Tabela 6.12 indicam que, como observado nos resultados dos experimentos para o problema *Knapsack*, a heurística adaptada produz resultados inferiores quando as instâncias resolvidas são mais simples. Isso nos indica que, nesses casos, a geração das novas ordenações da lista de itens a serem empacotados resulta em uma maior exploração do espaço de soluções, quando comparado com a heurística adaptada, que tenta otimizar soluções com as compactações de recipientes específicos. Uma última característica a ser comparada é o tempo de execução da heurística adaptada, que é maior devido o tempo gasto para se encerrar a compactação dos recipientes ao final do processo.

Como especificado, os resultados obtidos da literatura são os apresentados no trabalho de Terashima-Marín *et al.* [41]. Consideramos apenas os resultados obtidos para as instâncias irregulares, sendo estes obtidos por um conjunto básico de heurísticas de empacotamento.

Os experimentos das heurísticas apresentados por Terashima-Marín *et al.* para resolução do problema *Bin Packing* estão configurados da seguinte forma: foi utilizada uma combinação de 40 heurísticas de empacotamento para resolver cada uma das instâncias do problema, por um período de aproximadamente 12 horas. Dentre todas as heurísticas, adotamos os valores obtidos pela chamada "Filler", por apresentar os melhores resultados (maior porcentagem de resultados ótimos). O conjunto das instâncias utilizado por Terashima-Marín *et al.* foi dividido de diversas formas, com a separação em conjuntos de treinamento e teste. Apesar disso, tal divisão consiste em intercambiar tais conjuntos, de forma que os resultados entre os respectivos pares resulta no teste de todas as instâncias.

Tabela 6.12: Resultados das heurísticas propostas para o problema *Bin Packing*.

Nome	OPT	Heurística Proposta – Básica										Heurística Proposta – Adaptada									
		Min.	Máx.	Méd.	Tempo	Bins Extras						Min.	Máx.	Méd.	Tempo	Bins Extras					
						0	1	2	3	4	≥5					0	1	2	3	4	≥5
Fu	*	2	2	2.00	120.00	100.00	0.00	0.00	0.00	0.00	0.00	2	2	2.00	121.10	100.00	0.00	0.00	0.00	0.00	0.00
A	3	4	4	4.00	120.00	0.00	100.00	0.00	0.00	0.00	0.00	4	5	4.01	127.39	0.00	99.00	1.00	0.00	0.00	0.00
B	10	10	13	11.17	115.91	8.00	67.33	24.33	0.33	0.00	0.00	10	13	11.82	126.98	0.33	29.67	57.33	12.67	0.00	0.00
C	6	7	8	7.53	120.00	0.00	46.67	53.33	0.00	0.00	0.00	7	9	7.81	132.74	0.00	20.00	79.33	0.67	0.00	0.00
D	3	4	5	4.55	120.79	0.00	45.33	54.67	0.00	0.00	0.00	4	5	4.56	135.80	0.00	44.00	56.00	0.00	0.00	0.00
E	3	5	6	5.11	120.61	0.00	0.00	89.00	11.00	0.00	0.00	5	6	5.24	140.82	0.00	0.00	76.33	23.67	0.00	0.00
F	2	3	3	3.00	120.00	0.00	100.00	0.00	0.00	0.00	0.00	3	3	3.00	125.72	0.00	100.00	0.00	0.00	0.00	0.00
G	*	12	16	14.28	120.01	2.67	15.33	44.00	27.67	10.33	0.00	12	18	14.98	129.92	1.67	2.67	30.00	35.67	22.33	7.67
H	12	12	15	13.68	119.57	0.67	36.33	57.67	5.33	0.00	0.00	13	16	14.32	128.25	0.00	8.33	53.67	36.00	2.00	0.00
I	3	4	4	4.00	120.33	0.00	100.00	0.00	0.00	0.00	0.00	4	4	4.00	126.44	0.00	100.00	0.00	0.00	0.00	0.00
J	4	5	6	5.07	120.65	0.00	93.00	7.00	0.00	0.00	0.00	5	6	5.12	131.81	0.00	88.33	11.67	0.00	0.00	0.00
K	6	7	8	7.79	120.05	0.00	21.33	78.67	0.00	0.00	0.00	7	9	7.91	133.14	0.00	10.67	87.67	1.67	0.00	0.00
L	3	4	5	4.34	120.00	0.00	66.00	34.00	0.00	0.00	0.00	4	6	4.82	129.52	0.00	23.33	71.67	5.00	0.00	0.00
M	5	6	8	6.89	120.04	0.00	13.00	85.00	2.00	0.00	0.00	6	9	7.30	132.94	0.00	3.00	65.33	30.67	1.00	0.00
N	2	3	3	3.00	120.46	0.00	100.00	0.00	0.00	0.00	0.00	3	3	3.00	128.07	0.00	100.00	0.00	0.00	0.00	0.00
O	7	7	9	7.71	88.44	29.33	70.00	0.67	0.00	0.00	0.00	7	10	7.91	115.46	21.00	68.00	9.67	1.33	0.00	0.00
P	8	9	12	10.47	120.43	0.00	1.00	54.67	40.67	3.67	0.00	10	12	10.71	143.71	0.00	0.00	35.67	58.00	6.33	0.00
Q	15	16	18	17.36	120.81	0.00	2.33	59.33	38.33	0.00	0.00	17	20	18.06	133.98	0.00	0.00	16.00	62.33	21.00	0.67
R	9	10	12	11.13	120.39	0.00	0.33	86.33	13.33	0.00	0.00	11	13	11.31	142.16	0.00	0.00	69.00	30.67	0.33	0.00
					Média	**2.44**	**48.69**	**40.41**	**7.69**	**0.78**	**0.00**				*Média*	**1.46**	**38.65**	**39.94**	**16.54**	**2.94**	**0.46**

* Valor não disponível.

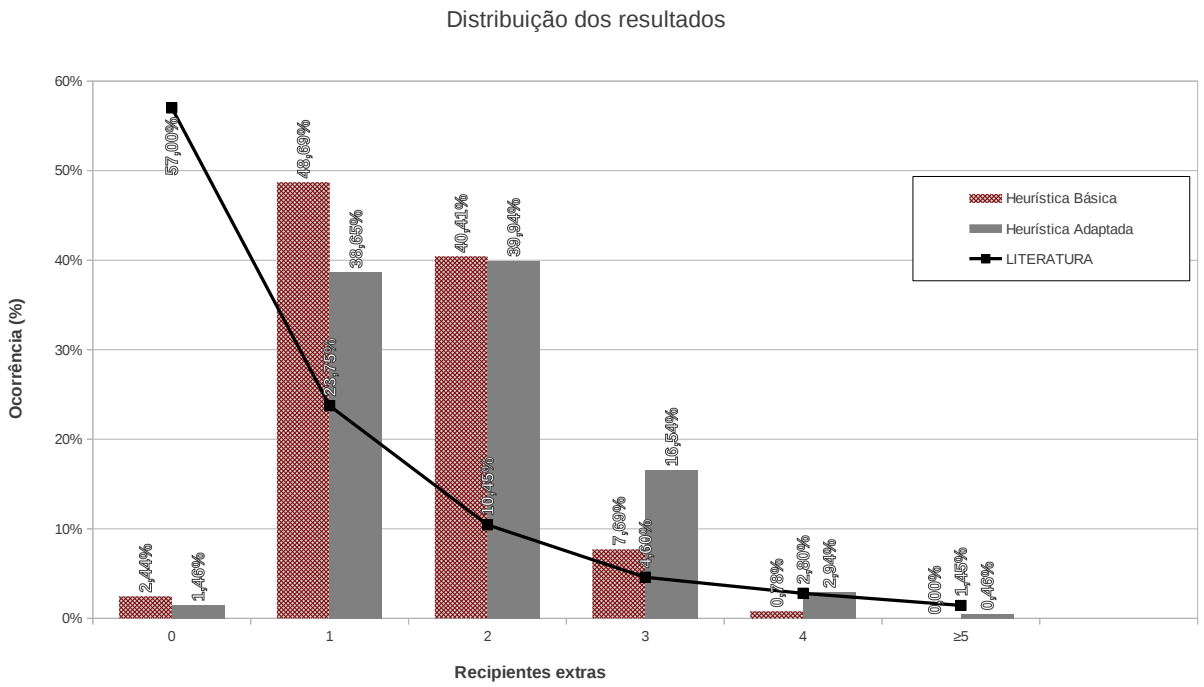


Figura 6.3: Comparação entre os resultados obtidos pelas heurísticas para o problema *Bin Packing*.

Logo, para comparação entre as heurísticas, utilizamos os valores médios referentes aos experimentos I e II obtidos pela heurística "Filler", que podem ser vistos na Figura 6.3.

A Figura 6.3 nos indica que os resultados obtidos por Terashima-Marín *et al.* são bastante superiores, quando comparamos os valores encontrados por sua heurística, referentes a solução ótima (0 recipiente extra), com os valores encontrados por nossas heurísticas, para este mesmo caso. Apesar disso, devemos levar alguns pontos em consideração. Primeiro, Terashima-Marín *et al.* não informam a quantidade executada de experimentos para qualquer heurística utilizada ao resolver uma instância do problema, além de não detalharem as configurações de ambiente para seus algoritmos. Diante disso, o tempo gasto por cada heurística para obter uma solução poderá ser maior, permitindo que melhores soluções possam ser buscadas. Outro ponto leva em conta a forma de apresentação dos resultados, que faz refência apenas à melhor solução encontrada. Isso dificulta a comparação entre as heurísticas, por não sabermos o comportamento de execução padrão do algoritmo com as referidas instâncias. Assim, a fixação dos parâmetros para a realização dos experimentos em nosso trabalho foi feita de forma a facilitar as comparações com novas heurísticas desenvolvidas.

Análise de desempenho do AG

Um resumo da contribuição do AG no processo de otimização dos problemas de empacotamento pode ser observado nas Figuras 6.4 e 6.5, em que apresentamos a evolução da otimização da solução para duas instâncias da base ESICUP (shirts e poly4b). Nesse caso, utilizamos os mesmos resultados de otimização obtidos do experimento executado, considerando a heurística básica proposta para resolver o problema *Strip Packing*.

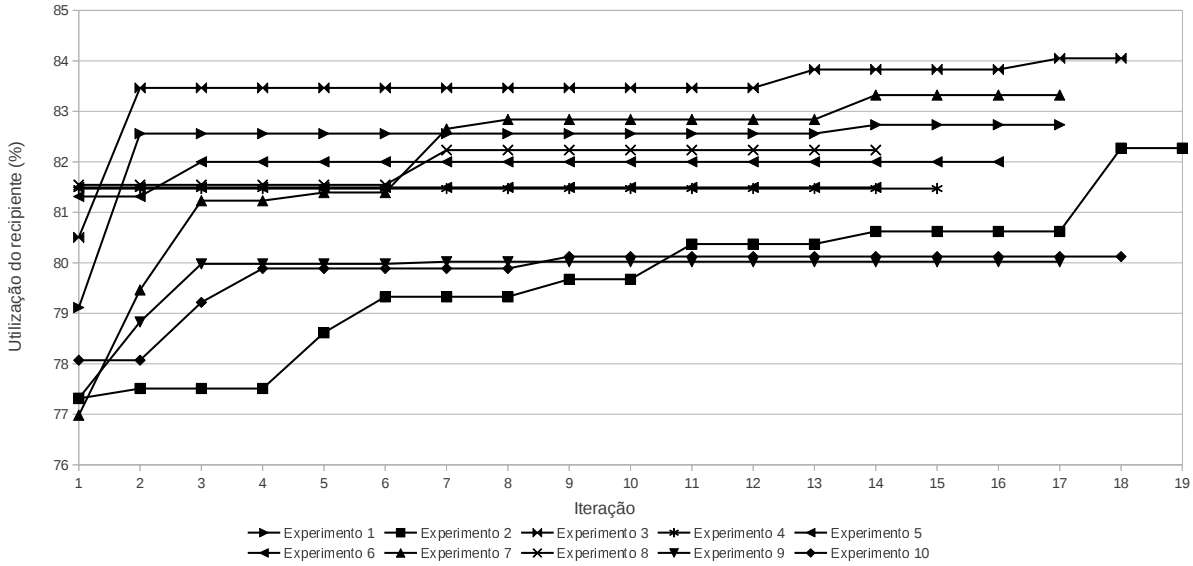


Figura 6.4: Evolução da otimização da instância shirts para o problema *Strip Packing*, para cada um dos experimentos realizados.

No geral, notamos alguns pontos importantes da referida otimização. Na maior parte dos casos, o início do processo de otimização tende a gerar a maior evolução do aproveitamento das soluções encontradas, sendo que, após a metade do tempo, esse comportamento é menos significativo. Apesar de acontecer raramente, há situações em que a estabilização da solução em um mínimo local já acontece na primeira iteração, de forma que, nem aplicando-se todas as estratégias de otimização apresentadas, a solução não conseguiu encontrar qualquer melhoria. Uma possível explicação para este fato seria ao pequeno número de iterações do algoritmo proposto. Como tais instâncias são bastante complexas, o algoritmo não conseguiu superar 24 iterações. Essa é uma desvantagem de se fixar um período de tempo para este tipo de problema, pois, devido a complexidade da instância otimizada, acaba-se limitando a exploração do espaço de soluções pelas metaheurísticas aplicadas. Para instâncias mais simples, o número de iterações é muito maior, e raramente isso acontece.

Todavia, podemos notar que o comportamento do AG é interessante, apesar de toda

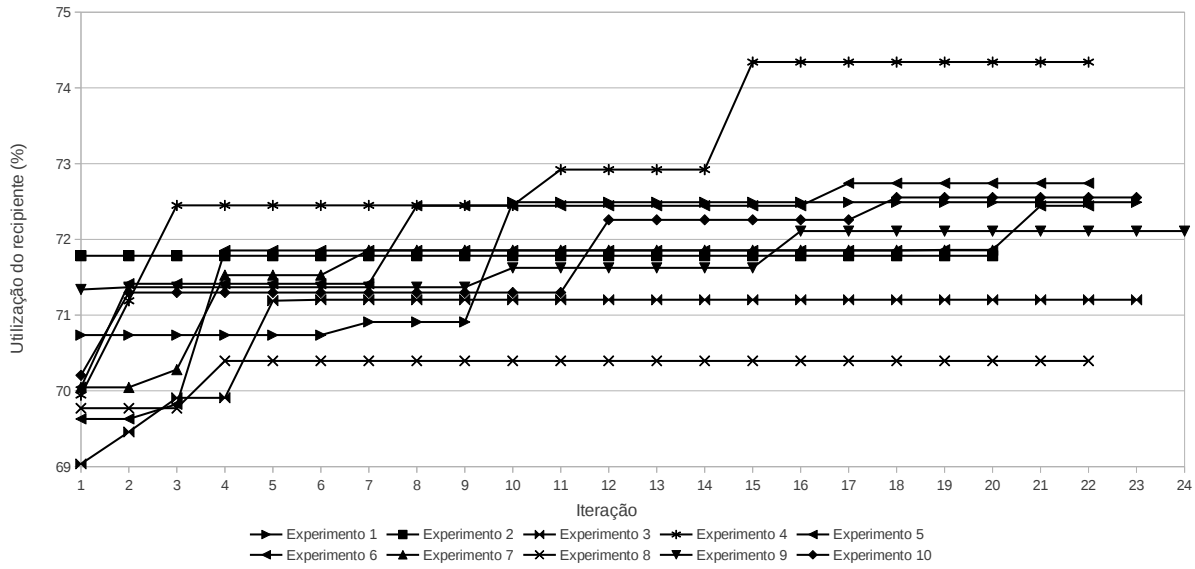


Figura 6.5: Evolução da otimização da instância poly4b para o problema *Strip Packing*, para cada um dos experimentos realizados.

sua robustez estar atrelada e depender diretamente da heurística de empacotamento adotada.

Capítulo 7

Conclusões

Esse capítulo apresenta um resumo do trabalho desenvolvido, com destaques às contribuições alcançadas. São apresentados, também, os novos objetivos a se alcançar com a continuação do seu desenvolvimento.

Este trabalho apresentou uma nova heurística para problemas de empacotamento com itens irregulares, utilizando uma abordagem construtiva da solução. A ideia básica foi utilizar a meta-heurística Algoritmo Genético para guiar o processo de construção de novas soluções, ao resolver os problemas *Strip Packing*, *Knapsack* e *Bin Packing*, utilizando itens irregulares. Os resultados mostraram que apesar da utilização de meta-heurísticas de otimização ser uma alternativa interessante, a escolha de uma boa heurística de empacotamento e de um tratamento geométrico ideal de todas as estruturas utilizadas são fundamentais para se gerar bons resultados, quando são considerados itens irregulares.

Foi proposta, também, uma heurística que faz a integração entre os algoritmos de problemas de empacotamento, de forma que o algoritmo para o problema *Strip Packing*, quando adaptado na heurística *First-Fit*, atua como um algoritmo de compactação, a fim de gerar soluções mais otimizadas para os problemas *Bin Packing* e *Knapsack*. Pelos resultados, a heurística de integração de algoritmos proposta se mostrou como uma opção interessante nos casos em que a instância a ser resolvida é mais complexa. Apesar disso, a flexibilidade proporcionada pela nova heurística acaba sendo interessante no sentido de que qualquer heurística que venha a ser desenvolvida para o problema *Strip Packing* possa ser utilizada como um novo mecanismo de compactação.

Em relação aos geradores de NFPs, a discussão acerca das técnicas e dos resultados obtidos por cada algoritmo implementado irá auxiliar outros pesquisadores na escolha de métodos de verificação de sobreposição para problemas de empacotamento com itens

irregulares.

Como trabalhos futuros, planejamos testar novas ferramentas de verificação de sobreposição, como as "Funções-Phi" (Chernov *et al.* [12]) e outras abordagens da soma de Minkowski, a fim de verificar seus desempenhos, quando utilizamos itens irregulares. Ainda considerando a estrutura dos NFPs, um trabalho futuro interessante é estender os algoritmos geradores implementados para que trabalhem com polígonos simples com buracos, generalizando o método para todo tipo de polígono simples. Em relação às heurísticas, buscaremos novas abordagens de empacotamento de itens, com e sem uso de NFPs, com o objetivo de analisar o impacto das meta-heurísticas com diferentes heurísticas de empacotamento. Por fim, iremos desenvolver novas abordagens de solução para os problemas de empacotamento, focando em heurísticas que geram soluções por meio da redução de sobreposição, além de buscarmos aplicações práticas para as heurísticas implementadas, a fim de resolver problemas reais de indústrias do setor de corte e empacotamento.

Referências Bibliográficas

[1] M. Adamowicz and A. Albano. Nesting two-dimensional shapes in rectangular modules. *Computer-Aided Design*, 8(1):27 – 33, 1976.

[2] J. A. Bennell and K. A. Dowsland. Hybridising tabu search with optimisation techniques for irregular stock cutting. *Management Science*, 47(8):1160–1172, 2001.

[3] J. A. Bennell and J. F. Oliveira. The geometry of nesting problems: A tutorial. *European Journal of Operational Research*, 184(2):397 – 415, 2008.

[4] J. A. Bennell and J. F. Oliveira. A tutorial in irregular shape packing problems. *Journal of the Operational Research Society*, 60:S93–S105(13), 2009.

[5] J. A. Bennell and X. Song. A comprehensive and robust procedure for obtaining the nofit polygon using minkowski sums. *Comput. Oper. Res.*, 35(1):267–281, January 2008.

[6] J. A. Bennell and X. Song. A beam search implementation for the irregular shape packing problem. *Journal of Heuristics*, 16:167–188, 2010.

[7] E. K. Burke, R. S. R. Hellier, G. Kendall, and G. Whitwell. A New Bottom-Left-Fill Heuristic Algorithm for the Two-Dimensional Irregular Packing Problem. *OPERATIONS RESEARCH*, 54(3):587–601, 2006.

[8] E. K. Burke, R. S. R. Hellier, G. Kendall, and G. Whitwell. Complete and robust no-fit polygon generation for the irregular stock cutting problem. *European Journal of Operational Research*, 179(1):27–49, 2007.

[9] E. K. Burke, R. S. R. Hellier, G. Kendall, and G. Whitwell. Irregular packing using the line and arc no-fit polygon. *Oper. Res.*, 58:948–970, July 2010.

[10] CGAL, Computational Geometry Algorithms Library, version 4.0, 4.0-2012. http://www.cgal.org.

[11] B. Chazelle and D. P. Dobkin. Optimal convex decompositions. *Computational Geometry*, pages 63–133, 1985.

[12] N. Chernov, Y. Stoyan, and T. Romanova. Mathematical model and efficient algorithms for object packing problem. *Computational Geometry*, 43(5):535–553, 2010.

[13] R Cunninghame-Green. Geometry, shoemaking and the milk tray problem. *New Scientist*, 12(1667):50–50, 1989.

[14] A. M. Del Valle, T. A. de Queiroz, F. K. Miyazawa, and E. C. Xavier. Heuristics for two-dimensional knapsack and cutting stock problems with items of irregular shape. *Expert Systems with Applications*, 2012.

[15] K. A. Dowsland and W. B. Dowsland. Solution approaches to irregular nesting problems. *European Journal of Operational Research*, 84(3):506 – 521, 1995.

[16] H. Dyckhoff. A typology of cutting and packing problems. *European J. Operational Research*, 44:145–159, 1990.

[17] J. Egeblad, B. K. Nielsen, and A. Odgaard. Fast neighborhood search for two- and three-dimensional nesting problems. *European Journal of Operational Research*, 183(3):1249 – 1266, 2007.

[18] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, Natural Computing Series, 2007.

[19] M. R. Garey and D. S. Johnson. Computers and intractability: a guide to the theory of np-hardness, 1979.

[20] P. K. Ghosh. An algebra of polygons through the notion of negative shapes. *CVGIP: Image Underst.*, 54:119–144, June 1991.

[21] P. K. Ghosh. A unified computational framework for minkowski operations. *Computers and Graphics*, 17(4):357 – 378, 1993.

[22] A. M. Gomes and J. F. Oliveira. Solving irregular strip packing problems by hybridising simulated annealing and linear programming. *European Journal of Operational Research*, 171(3):811 – 829, 2006.

[23] B. Grünbaum, V. Klee, M. A. Perles, and G. C. Shephard. *Convex polytopes*, volume 2. Springer, Londres, 1967.

[24] J. H. Holland. *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA, 1992.

[25] E. Hopper. *Two Dimensional Packing utilising evolutionary algorithms and other meta-heuristic methods*. PhD thesis, University of Wales, Cardiff, 2000.

[26] http://paginas.fe.up.pt/∼esicup/. Euro special interest group on cutting and packing, January 2013.

[27] T. Imamichi, M. Yagiura, and H. Nagamochi. An iterated local search algorithm based on nonlinear programming for the irregular strip packing problem. *Discrete optimization*, 6(4):345–361, 2009.

[28] M. Konopasek. Mathematical treatments of some apparel marking and cutting problems. *US Department of Commerce Report*, 99(26):90857–10, 1981.

[29] S. C. H. Leung, Y. Lin, and D. Zhang. Extended local search algorithm based on nonlinear programming for two-dimensional irregular strip packing problem. *Computers & Operations Research*, 39(3):678 – 686, 2012.

[30] Z. Li and V. Milenkovic. Compaction and separation algorithms for non-convex polygons and their applications. *European Journal of Operational Research*, 84(3):539–561, 1995.

[31] E. López-Camacho, H. Terashima-Marín, and P. Ross. Defining a problem-state representation with data mining within a hyper-heuristic model which solves 2d irregular bin packing problems. In *Advances in Artificial Intelligence–IBERAMIA 2010*, pages 204–213. Springer, 2010.

[32] A. Mahadevan. Optimization in computer-aided pattern packing. *Dissertation Abstracts International Part B: Science and Engineering[DISS. ABST. INT. PT. B-SCI. & ENG.]*,, 46(2), 1985.

[33] T. C. Martins and M. S. G. Tsuzuki. Simulated annealing applied to the irregular rotational placement of shapes over containers with fixed dimensions. *Expert Systems with Applications*, 37(3):1955–1972, 2010.

[34] L. R. Mundim and T. A. de Queiroz. A hybrid heuristic for the 0–1 knapsack problem with items of irregular shape. In *Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En*, pages 1–6. IEEE, 2012.

[35] B. K. Nielsen and A. Odgaard. Fast neighborhood search for the nesting problem. Technical Report 03/03, Department of Computer Science, University of Copenhagen, Universitetsparken 1, DK-2100 Copenhagen, Denmark, 2003.

[36] J. F. Oliveira, A. M. Gomes, and J. S. Ferreira. Topos - a new constructive algorithm for nesting problems. *OR Spectrum*, 22:263–284, 2000.

[37] F. P. Preparata and M. I. Shamos. *Computational geometry: an introduction. 1985.* Springer-Verlag, 1985.

[38] G. D. Ramkumar. An algorithm to compute the minkowski sum outer-face of two simple polygons. In *Proceedings of the twelfth annual symposium on Computational geometry*, pages 234–241. ACM, 1996.

[39] A. K. Sato, T. C. Martins, and M. S. G. Tsuzuki. An algorithm for the strip packing problem using collision free region and exact fitting placement. *Computer-Aided Design*, 44(8):766–777, 2012.

[40] M. A Shalaby and M. Kashkoush. A particle swarm optimization algorithm for a 2-d irregular strip packing problem. *American Journal of Operations Research*, 3:268–278, 2013.

[41] H. Terashima-Marín, P. Ross, C. J. Farías-Zárate, E. López-Camacho, and M. Valenzuela-Rendón. Generalized hyper-heuristics for solving 2d regular and irregular packing problems. *Annals of Operations Research*, 179:369–392, 2010.

[42] S. Umetani, M. Yagiura, S. Imahori, T. Imamichi, K. Nonobe, and T. Ibaraki. Solving the irregular strip packing problem via guided local search for overlap minimization. *International Transactions in Operational Research*, 16(6):661–683, 2009.

[43] G. Wäscher, H. Haußner, and H. Schumann. An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183(3):1109–1130, 2007.