**Estudo de métodos de solução para problemas de corte de
itens irregulares em recipientes irregulares**

Felipe Augusto Aureliano

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências
de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Felipe Augusto Aureliano

Estudo de métodos de solução para problemas de corte de itens irregulares em recipientes irregulares

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Marina Andretta

USP – São Carlos
Agosto de 2017

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

A634e	<p>Aureliano, Felipe Augusto</p> <p>Estudo de métodos de solução para o problema de corte de itens irregulares em recipientes irregulares / Felipe Augusto Aureliano; orientadora Marina Andretta. - São Carlos - SP, 2017.</p> <p>89 p.</p> <p>Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2017.</p> <p>1. problemas de corte e empacotamento. 2. problemas de corte de itens irregulares. 3. heurísticas. I. Andretta, Marina, orient. II. Título.</p>
-------	--

Felipe Augusto Aureliano

**Study of solution methods for the irregular bin packing
problem**

Master dissertation submitted to the Instituto de
Ciências Matemáticas e de Computação - ICMC-
USP, in partial fulfillment of the requirements for the
degree of the Master Program in Computer Science
and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Profa. Dra. Marina Andretta

**USP – São Carlos
August 2017**

Aos meus pais, Paulo e Mara.

AGRADECIMENTOS

Primeiramente, aos meus pais, Paulo e Mara, por acreditarem na importância dos estudos e me apoiarem em cada decisão tomada. Também, aos meus demais familiares por todo o incentivo.

Aos meus amigos, por toda alegria a mim proporcionada, e por compreenderem os momentos em que estive menos presente.

À todos os professores e alunos do Laboratório de Otimização com os quais tive a oportunidade de conviver durante todo este período. Palavras não são suficientes para descrever o quão grato eu sou pelo amparo e incentivo vindos de cada um.

Agradeço muito à minha orientadora Marina Andretta por toda paciência, colaboração e disponibilidade para reuniões de emergência.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro, sem o qual não teria sido possível o desenvolvimento deste trabalho.

“Em primeiro lugar vem a dedicação, depois a habilidade.”
(Leonardo Da Vinci)

RESUMO

AURELIANO, F. A.. **Estudo de métodos de solução para o problema de corte de itens irregulares em recipientes irregulares**. 2017. 89 f. Dissertação (Mestrado em em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

Dentro da classe de problemas de corte e empacotamento, existem os problemas de corte de itens irregulares (não-circulares e não-retangulares), os quais visam determinar um arranjo ótimo de objetos irregulares menores (itens), sem sobreposição, dentro de objetos maiores (recipientes) a fim de atender a uma demanda. Possuem grande importância prática, uma vez que surgem em vários tipos de indústrias, como a têxtil, a de móveis e a de calçados, por exemplo. Entre estes problemas, ainda temos o chamado problema de corte de itens irregulares em recipientes, no qual estes últimos são fechados, isto é, possuem dimensões fixas, podendo ser retangulares ou irregulares. Neste caso, o objetivo é arranjar todos os itens de modo a utilizar o menor número possível de recipientes. A estes problemas, uma outra restrição ainda pode ser adicionada: os recipientes podem ter defeitos, isto é, áreas onde não pode ser posicionado qualquer item, e regiões com diferentes níveis de qualidade, chamadas de zonas de qualidades, em que apenas determinados itens podem ser alocados. Neste trabalho, portanto, introduzimos um conjunto de heurísticas construtivas para a resolução do problema de corte de itens irregulares em recipientes irregulares com defeitos e zonas de qualidades. Os experimentos computacionais foram realizados utilizando um conjunto com 15 instâncias adaptadas de outro problema de corte de itens irregulares, uma vez que não encontramos instâncias disponíveis na literatura para o problema abordado neste trabalho. Os resultados mostraram que todos os métodos são capazes de resolver o problema em um tempo computacional considerado baixo, sendo que alguns deles apresentam melhor desempenho que outros.

Palavras-chave: problemas de corte e empacotamento, problemas de corte de itens irregulares, heurísticas.

ABSTRACT

AURELIANO, F. A.. **Estudo de métodos de solução para o problema de corte de itens irregulares em recipientes irregulares**. 2017. 89 f. Dissertação (Mestrado em em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

Within the class of cutting and packing problems, there are some problems known as nesting problems, which aim to determine an optimal arrangement of smaller irregular objects (items), without overlap, inside larger objects (bins) in order to attend a demand. They have practical importance, since they arise in many types of industries, such as textiles, furniture and footwear, for example. Among these problems, we still have the so-called irregular bin packing problem in which the bins are closed, that is, they have fixed dimensions, and may be rectangular or irregular. In this case, the goal is to arrange all items in order to use the least amount of bins. To these problems, another constraint can still be added: the bins may have defects, that is, areas where no item can be placed, and different levels of quality, called quality zones, where only specific items can be allocated. In this work, therefore, we introduce a set of constructive heuristics to solve the irregular bin packing problem in which the bins have defects and quality zones. The computational experiments were carried out using a set of 15 instances adapted from another nesting problem, since we did not find instances available in the literature for the problem addressed in this work. The results showed that all methods can solve the problem in a low computational time, and also that some of them perform better than others.

Key-words: cutting and packing problems, nesting problems, heuristics.

SUMÁRIO

1	INTRODUÇÃO	17
2	DEFINIÇÃO DO PROBLEMA	19
3	REVISÃO DA LITERATURA	21
3.1	Representação dos itens e dos recipientes	21
3.2	Verificação de sobreposição de itens	25
3.2.1	<i>Rasterização</i>	25
3.2.2	<i>Trigonometria direta</i>	27
3.2.3	<i>Função ϕ</i>	29
3.2.4	<i>Nofit polygon</i>	30
3.3	Posicionamento de itens no interior de um recipiente	32
3.4	Métodos de resolução	35
3.4.1	<i>Modelos e métodos exatos</i>	35
3.4.1.1	<i>Corte de itens irregulares em faixa</i>	35
3.4.1.2	<i>Corte de itens irregulares em recipientes fechados</i>	37
3.4.2	<i>Métodos heurísticos</i>	38
3.4.2.1	<i>Problemas de corte de itens irregulares em faixa</i>	39
3.4.2.2	<i>Problemas de corte de itens irregulares em recipientes fechados</i>	40
4	MÉTODOS DE RESOLUÇÃO PROPOSTOS	45
4.1	Heurística construtiva básica	45
4.2	Variações das heurísticas	51
4.3	Implementação	54
4.3.1	<i>Dados de entrada</i>	54
4.3.2	<i>Dados de saída</i>	58
5	EXPERIMENTOS COMPUTACIONAIS	59
5.1	Instâncias	59
5.2	Resultados Numéricos	62
6	CONCLUSÕES E TRABALHOS FUTUROS	67
	Referências	69

APÊNDICE A	INSTÂNCIAS	73
A.1	Albano-z	73
A.2	Dagli-z	74
A.3	Dighe1-z	74
A.4	Dighe2-z	75
A.5	Fu-z	76
A.6	Jakobs1-z	76
A.7	Jakobs2-z	77
A.8	Mao-z	78
A.9	Marques-z	78
A.10	Shapes0-z	79
A.11	Shapes1-z	79
A.12	Shapes2-z	80
A.13	Shirts-z	81
A.14	Swim-z	81
A.15	Trousers-z	82
APÊNDICE B	MELHORES SOLUÇÕES	83

INTRODUÇÃO

Problemas de Corte e Empacotamento (*cutting and packing problems*) são problemas de otimização combinatória que têm por objetivo determinar um arranjo ótimo de objetos menores (itens) no interior de objetos maiores (recipientes) visando a minimização de desperdícios. Esta última, além de interessante do ponto de vista sustentável, contribui para a redução dos custos de produção. Estes problemas têm em comum a existência de, no mínimo, duas restrições: uma para evitar que os itens se sobreponham entre si e outra para garantir que os mesmos estejam inteiramente contidos no recipiente.

No contexto destes problemas, existem aqueles nos quais os itens e/ou recipientes possuem formas irregulares (não-retangulares e não-circulares). Conhecidos como problemas de corte de itens irregulares, ou problemas de *nesting*, geralmente estão presentes em diversos ramos da indústria em que se necessita cortar ou encaixar múltiplos itens irregulares, como, por exemplo, na indústria têxtil, de móveis, de calçados e metalmecânica.

A principal dificuldade encontrada na resolução de problemas de corte de itens irregulares, uma vez que apresentam uma componente geométrica mais complexa, é evitar a sobreposição dos itens entre si. Para tratar esta questão, existem quatro ferramentas na literatura bastante utilizadas, sendo elas, rasterização, trigonometria direta, *nofit polygon* e função ϕ (BENNELL; OLIVEIRA, 2008). Segundo Bennell e Oliveira (2008), ao optar por qual ferramenta utilizar, não se deve considerar apenas a eficiência, mas também o quão difíceis são de serem implementadas de forma robusta.

Entre os problemas de corte de itens irregulares existe o problema de corte de itens irregulares em faixa (*irregular strip packing problem*), bastante conhecido na literatura, cujo objetivo é arranjar um conjunto de itens irregulares em um único recipiente retangular de largura fixa e comprimento “infinito” de modo que este último seja mínimo. Há também aqueles problemas em que o recipiente, que pode ser retangular ou irregular, é fechado, isto é, possui dimensões fixas. Este é o caso do problema de corte de itens irregulares em recipientes (*irregular*

bin packing problem), com o qual trabalhamos em nossa pesquisa. Neste caso, o objetivo é empacotar um conjunto de itens utilizando a menor quantidade possível de recipientes.

Além da forma irregular, os recipientes ainda podem apresentar defeitos, isto é, áreas onde não pode ser posicionado nenhum item, e também regiões com diferentes níveis de qualidades, chamadas de zonas de qualidades, em que apenas determinados itens podem ser alocados. O couro é um exemplo clássico de recipiente que apresenta zonas de qualidades, e tem sido utilizado por alguns autores como objeto de estudos em problemas de corte de itens irregulares (HEISTERMANN; LENGAUER, 1995; CRISPIN *et al.*, 2005; LEE; MA; CHENG, 2008; ZHANG; YANG, 2009; ALVES *et al.*, 2012a; ALVES *et al.*, 2012b; BALDACCI *et al.*, 2014; PINTO *et al.*, 2016).

O objetivo deste trabalho de mestrado foi, portanto, resolver o problema de corte de itens irregulares em recipientes irregulares com defeitos e zonas de qualidades. Para isso, desenvolvemos um conjunto de métodos heurísticos construtivos, os quais diferem-se uns dos outros em relação a algumas estratégias características de métodos heurísticos para problemas de corte e empacotamento. De modo a validar os métodos, realizamos uma série de experimentos computacionais utilizando um conjunto de 15 instâncias adaptadas do problema de corte de itens irregulares em faixa, visto que não encontramos instâncias disponíveis na literatura para o problema abordado neste trabalho.

O trabalho está estruturado conforme a seguir. No Capítulo 2, definimos o problema de pesquisa. No Capítulo 3, fazemos uma revisão da literatura sobre problemas de corte de itens irregulares. No Capítulo 4, apresentamos os métodos desenvolvidos para a resolução do problema abordado. No Capítulo 5, introduzimos as instâncias utilizadas nos experimentos computacionais, assim como os resultados obtidos. Por fim, no Capítulo 6, trazemos algumas considerações finais sobre o trabalho e indicamos algumas sugestões para trabalhos futuros.

DEFINIÇÃO DO PROBLEMA

Neste trabalho, resolvemos o problema bidimensional de corte **ou** empacotamento de itens irregulares em recipientes (*irregular bin packing problem*), que pode ser definido da seguinte forma: dadas uma lista de itens irregulares $I = (1, 2, \dots, n)$ e uma lista de recipientes retangulares ou irregulares $R = (1, 2, \dots, m)$, o objetivo é empacotar os n itens de I utilizando a menor quantidade possível de recipientes da lista R . O empacotamento deve ser feito de modo que os itens fiquem inteiramente contidos nos recipientes e não se sobreponham entre si.

Além de irregulares, consideramos que os recipientes também podem apresentar defeitos, sobre os quais fica proibido o posicionamento de qualquer item, tais como buracos, por exemplo, e regiões com diferentes níveis de qualidades, chamadas de zonas de qualidades, em que apenas determinados itens podem ser posicionados. Consideramos, também, que um item só pode ser alocado sobre uma determinada zona de qualidade, caso a qualidade desta zona seja igual ou superior à qualidade mínima exigida para este item. Assim, proibimos que o mesmo seja composto por qualquer qualidade considerada indesejável (inferior) para ele.

O problema de corte de itens irregulares em recipientes irregulares com defeitos e zonas de qualidades surge principalmente em indústrias cujos produtos são fabricados a partir do couro, tais como mobílias, sapatos, vestuários e assentos de carros (HEISTERMANN; LENGAUER, 1995; CRISPIN *et al.*, 2005; ALVES *et al.*, 2012a; ALVES *et al.*, 2012b; BALDACCI *et al.*, 2014; PINTO *et al.*, 2016). Este problema possui bastante relevância do ponto de vista ambiental, uma vez que, quanto maior for o aproveitamento do couro, menor será o descarte de sobras. A Figura 1 ilustra um exemplo de empacotamento de itens irregulares em couro.

Figura 1 – Exemplo de um empacotamento de itens irregulares em couro.

Fonte: Adaptada de [Baldacci et al. \(2014\)](#).

A motivação pela escolha deste problema surgiu a partir de dois fatos: sua relevância, uma vez que, como já citado, o problema de corte e empacotamento de itens irregulares em recipientes com defeitos e zonas de qualidades surge em vários setores da indústria; e a pequena quantidade de trabalhos existentes na literatura que o resolvem, o que ocorre devido à sua dificuldade. Pretendemos com este trabalho, portanto, contribuir cientificamente para as pesquisas na área implementando novos métodos de resolução para o problema.

REVISÃO DA LITERATURA

3.1 Representação dos itens e dos recipientes

Nesta seção, discorreremos a princípio sobre diferentes possibilidades para representar os itens e, em seguida, os recipientes. Saber como representá-los de forma adequada é uma tarefa importante para a eficácia de qualquer método de resolução. É importante deixar claro que, ao longo do trabalho, eventualmente nos referimos aos itens e recipientes como objetos do problema.

A representação por malha é uma técnica bastante simples que consiste basicamente em utilizar uma malha de modo que o item cubra um conjunto de posições (SEGENREICH; BRAGA, 1986; BABU; BABU, 2001; BALDACCI *et al.*, 2014). Esta abordagem é capaz de representar desde itens mais simples, como por exemplo, retângulos (Figura 2(a)), até itens mais complexos, tais como aqueles que apresentam concavidades e/ou curvas (Figura 2(b)). Observe, ainda na Figura 2(b), que a desvantagem desta técnica é a representação não-exata de arestas não-ortogonais e curvas. Vale também mencionar que quanto menor for o tamanho da unidade da malha, mais precisa se torna a representação dos itens.

Figura 2 – Representação de itens através de malhas.

Fonte: Elaborada pelo autor.

Uma outra abordagem muito utilizada na literatura e que apresenta maior precisão em relação à anterior é a representação por polígonos. Neste caso, cada item é representado por meio de um único polígono convexo ou não-convexo (HEISTERMANN; LENGAUER, 1995; ALVES *et al.*, 2012a; TOLEDO *et al.*, 2013; ELKERAN, 2013). Mais especificamente, quando um item possui as características de um polígono, podemos simplesmente considerá-lo como tal, obtendo, portanto, uma representação precisa (Figura 3). Por outro lado, determinados tipos de itens precisam ter sua forma aproximada a de um polígono, como é o caso daqueles que possuem curvas, como mostra o exemplo da Figura 4.

Figura 3 – Itens precisamente representados por polígonos.

Fonte: Elaborada pelo autor.

Figura 4 – Item representado por um polígono aproximado.

Fonte: Elaborada pelo autor.

Por fim, uma outra possibilidade ainda é representá-los através de conjuntos de polígonos convexos (CHERRI *et al.*, 2016). Mais especificamente, a ideia é, dado um item, decompô-lo em um conjunto de polígonos convexos. Esta técnica, na realidade, mostra-se interessante para tratar itens que apresentam concavidades e/ou buracos, como ilustra a Figura 5.

Figura 5 – Itens representados por conjuntos de polígonos convexos.

Fonte: Elaborada pelo autor.

Assim como os itens, os recipientes também podem ser representados por malhas ou polígonos. A representação por malha consiste em dividir o recipiente em áreas discretas, reduzindo, portanto, para finito o conjunto de posições factíveis (SEGENREICH; BRAGA, 1986; OLIVEIRA; FERREIRA, 1993; BABU; BABU, 2001; TOLEDO *et al.*, 2013; BALDACCI *et al.*, 2014). Observe na Figura 6 que, diferentemente dos itens, a representação dos recipientes

é aproximada para dentro, isto é, arestas não-ortogonais e curvas não são representadas. Isto ocorre porque não se pode considerar regiões que não existem de fato.

Figura 6 – Representação de um recipiente irregular R através de uma malha.

Fonte: Elaborada pelo autor.

Com relação à representação por polígonos, utilizamos a mesma ideia aplicada aos itens. Caso o recipiente já possua as características de um polígono, simplesmente o consideramos como tal, caso contrário, encontramos um polígono que o represente de forma aproximada (HEISTERMANN; LENGAUER, 1995; ALVES *et al.*, 2012a; ELKERAN, 2013). A Figura 7 ilustra um exemplo de uma representação aproximada do recipiente R da Figura 6.

Figura 7 – Representação aproximada do recipiente R por um polígono.

Fonte: Elaborada pelo autor.

Por fim, podemos também representar um recipiente irregular qualquer através de um retângulo com um conjunto de polígonos convexos (MARTINS; TSUZUKI, 2010). Mais especificamente, a ideia é tomar o menor retângulo que envolve completamente o recipiente e considerar as regiões entre sua borda e as arestas do retângulo como se fossem buracos (defeitos), que podem ser então aproximados por polígonos convexos (Figura 8). Dessa forma, só devemos garantir que não haja sobreposição entre os itens do problema e tais defeitos. Na Seção 3.2, apresentamos mais detalhes sobre a restrição de não-sobreposição.

Os recipientes podem conter buracos e outros tipos de defeitos, sobre os quais fica proibida a alocação de qualquer item. Além disso, podem também conter regiões com diferentes níveis de qualidades (chamadas de zonas de qualidades), em que apenas determinados itens podem ser posicionados. A decisão sobre quais zonas um item pode ou não sobrepor depende da qualidade exigida para aquele item. Por exemplo, caso seja ordenado que um item possua uma

Figura 8 – Representação do recipiente R segundo a abordagem utilizada por Martins e Tsuzuki (2010).

Fonte: Elaborada pelo autor.

qualidade específica (ou uma qualidade mínima), o mesmo não poderá ser cortado a partir de uma região do recipiente que possua qualquer outra qualidade (ou qualidade inferior à mínima). A Figura 9 ilustra um recipiente R' (adaptação do recipiente R) com um buraco e algumas zonas de qualidades.

Figura 9 – Recipiente com defeito e zonas de qualidades.

Fonte: Elaborada pelo autor.

Para lidar com os defeitos e zonas de qualidades, Baldacci *et al.* (2014) optaram por uma abordagem com malhas, na qual associaram a cada posição da malha do recipiente um valor que indica seu nível de qualidade. Tal valor pode variar em uma escala de 0 a um número inteiro não-negativo q_{max} , em que 0 representa a pior qualidade e q_{max} a qualidade máxima. O posicionamento de um item é considerado factível se cada posição da malha por ele ocupada possui qualidade igual ou superior à qualidade mínima exigida para o item.

Alves *et al.* (2012a) optaram por aproximar os defeitos e zonas de qualidades por polígonos e designaram a cada um deles um valor que representa seu nível de qualidade. Neste caso, foi utilizada uma escala que vai de A a D , em que A e D representam a melhor e pior qualidade, respectivamente. Para que o posicionamento de um item, que também pode ser composto por zonas de qualidades seja adequado, nenhuma de suas zonas pode ser posicionada sobre regiões do recipiente que possuam qualidade inferior.

3.2 Verificação de sobreposição de itens

Uma grande dificuldade encontrada em problemas de corte de itens irregulares é evitar a sobreposição entre itens. Devido a este fato, alguns pesquisadores têm se dedicado ao longo dos anos ao desenvolvimento e aperfeiçoamento de métodos para lidar com tal restrição. Quando optamos por representar os itens por malhas, o método mais citado na literatura é o da rasterização. Por outro lado, quando fazemos uso da representação poligonal (seja associando a um item um único polígono ou um conjunto de polígonos convexos), três métodos são facilmente encontrados, sendo eles: trigonometria direta, função ϕ e *nofit polygon*.

A seguir, apresentamos cada uma das quatro ferramentas citadas. Contudo, é interessante previamente mencionar que, para verificar a sobreposição entre dois itens representados por conjuntos de polígonos convexos, caso esta seja a abordagem escolhida, basta investigar se há sobreposição entre pelo menos um polígono convexo de um item e um polígono convexo do outro. Tendo isto em vista, discorreremos sobre os métodos, com exceção da rasterização, considerando que os itens são representados cada um por um único polígono.

3.2.1 Rasterização

Quando optamos por reduzir a complexidade geométrica dos objetos do problema a malhas, podemos utilizar o método da rasterização para verificar a sobreposição entre itens. Neste caso, as malhas passam a ser representadas por matrizes, cujas células recebem valores por meio de algum esquema de codificação, e a verificação de sobreposição entre dois itens acaba reduzindo-se à soma de duas matrizes e à análise dos valores resultantes. A seguir, explanamos o método de forma mais detalhada a partir de duas diferentes estratégias de codificação encontradas na literatura.

Para codificar um item, [Segenreich e Braga \(1986\)](#) utilizaram uma matriz na qual os algarismos 1 e 3 representam sua fronteira e interior, respectivamente, e o 0 denota posições vazias (Figura 10). Assim, dado um item a ser inserido, se ao adicionarmos sua matriz com a matriz do atual leiaute obtivermos alguma célula com valor igual ou superior a 4, isso significa que há mais de um item ocupando a posição em questão no leiaute, configurando, portanto, uma sobreposição (Figura 11). Quando apenas o valor 2 é obtido, isto significa que o item está em contato com outro ou a uma distância bem próxima, dependendo do tamanho definido para a unidade da malha.

[Oliveira e Ferreira \(1993\)](#) introduziram uma abordagem mais simples, porém também eficiente. Neste caso, é utilizada uma matriz binária para representar um item, em que o valor 1 indica a existência de item e o 0 denota espaços vazios (Figura 12). Sendo assim, uma sobreposição é identificada caso seja obtido algum valor superior a 1 após a adição das matrizes (Figura 13).

Figura 10 – Representação de um item segundo [Segenreich e Braga \(1986\)](#).

Fonte: Elaborada pelo autor.

Figura 11 – Sobreposição entre dois itens segundo a abordagem de [Segenreich e Braga \(1986\)](#).

Fonte: Elaborada pelo autor.

Figura 12 – Representação de um item segundo [Oliveira e Ferreira \(1993\)](#).

Fonte: Elaborada pelo autor.

Figura 13 – Sobreposição entre dois itens segundo a abordagem de [Oliveira e Ferreira \(1993\)](#).

Fonte: Elaborada pelo autor.

Como visto, o método da rasterização é bastante simples, uma vez que a sobreposição entre itens pode ser identificada simplesmente analisando os valores das células de uma matriz. Além disso, mover um item no recipiente de modo a eliminar infactibilidade é simplesmente uma questão de contar células na direção desejada.

3.2.2 Trigonometria direta

A trigonometria direta consiste de testes que envolvem as arestas e vértices de polígonos para identificar se há ou não sobreposição entre eles. Especificamente, dados dois polígonos, se ambos possuem arestas que se interceptam, ou se pelo menos um vértice de um deles pertence ao interior do outro, então os polígonos estão sobrepostos. Contudo, observe que, se os retângulos que envolvem dois polígonos não se sobrepõem, então os mesmos também não se sobrepõem. Ou seja, realizando este teste previamente, identificaríamos casos de não-sobreposição de forma mais rápida e simples. Sendo assim, [Bennell e Oliveira \(2008\)](#) propuseram a seguinte sequência de testes de modo a tornar o processo de verificação mais eficiente:

1. Se os retângulos envolventes de dois polígonos não se sobrepõem, então os polígonos não se sobrepõem (Figura 14(a)). Neste caso, pare; caso contrário, vá para o teste 2.
2. Se para cada par de arestas, em que cada uma pertence a um diferente polígono, seus respectivos retângulos envolventes não se sobrepõem, então os polígonos também não se sobrepõem (Figura 14(b)). Neste caso, pare; caso contrário, vá para o teste 3.
3. Considere todos os pares de arestas cujos respectivos retângulos envolventes se sobrepõem. Se pelo menos um destes pares se intercepta, então os polígonos estão sobrepostos (Figura 14(c)). Neste caso, pare; caso contrário, vá para o teste 4.
4. Se pelo menos um vértice de um polígono pertence ao interior do outro polígono, então os polígonos se sobrepõem (Figura 14(d)); caso contrário, não há sobreposição (Figura 14(e)).

Figura 14 – Exemplos de posições relativas entre dois polígonos identificadas pelos testes de 1 a 4.

Fonte: [Bennell e Oliveira \(2008\)](#).

Segundo [Bennell e Oliveira \(2008\)](#), os testes 1 e 2 são fáceis de realizar, uma vez que utilizam apenas comparações de coordenadas. [Preparata e Shamos \(1985\)](#) desenvolveram um método simples que pode ser usado no teste 4. Por fim, para verificar a intersecção entre arestas no teste 3, podemos utilizar a função D , cujo princípio vem da equação da distância de um ponto a uma reta. Introduzida por [Konopasek \(1981\)](#), a função D pode ser definida da seguinte forma:

$$D_{ABP} = ((X_A - X_B)(Y_A - Y_P) - (Y_A - Y_B)(X_A - X_P)).$$

O valor D_{ABP} fornecido pela função nos permite identificar a posição relativa entre uma aresta orientada AB e um ponto P . Ou seja,

- se $D_{ABP} < 0$, então o ponto P está no lado direito da reta suporte da aresta AB (Figura 15(a));
- se $D_{ABP} > 0$, então o ponto P está no lado esquerdo da reta suporte da aresta AB (Figura 15(b));
- por fim, se $D_{ABP} = 0$, então o ponto P pertence à linha suporte de AB e, possivelmente, à AB (Figura 15(c)).

Figura 15 – Interpretação da função D .

Fonte: Elaborada pelo autor.

Mais ainda, a função D pode ser utilizada para determinar a posição relativa entre duas arestas orientadas. Segundo [Mahadevan \(1984\)](#), a intersecção entre duas arestas orientadas AB e UV , representada geometricamente pela Figura 16, pode ser identificada pela função D da seguinte forma:

$$D_{ABU} \neq 0 \wedge D_{ABV} \neq 0 \wedge D_{ABU} \neq D_{ABV} \wedge D_{UVA} \neq 0 \wedge D_{UVB} \neq 0 \wedge D_{UVA} \neq D_{UVB}.$$

Figura 16 – Intersecção entre AB e UV .

Fonte: Elaborada pelo autor.

Deste modo, portanto, já conhecemos ferramentas o suficiente para realizar os testes propostos por [Bennell e Oliveira \(2008\)](#).

3.2.3 Função ϕ

Segundo [Bennell e Oliveira \(2008\)](#), a função ϕ é uma ferramenta inovadora e poderosa para tratar a restrição de não-sobreposição entre itens. Neste caso, para cada par de itens, deriva-se uma função ϕ específica cujo objetivo é representar todas as posições relativas entre eles. Até onde se sabe, o processo de derivação é feito manualmente através do uso de ferramentas da trigonometria.

A ideia é a seguinte: sejam A e B dois itens quaisquer. Se o valor da função ϕ_{AB} é maior do que zero, então A e B estão separados; se o valor de ϕ_{AB} é igual a zero, então A e B estão se tocando; por fim, se o valor de ϕ_{AB} é menor do que zero, então A e B sobrepõem-se. A Figura 17 ilustra graficamente a função ϕ entre dois itens representados por retângulos. A região definida em negrito indica todas as posições em que a função se anula e, consequentemente, indica todas as posições na quais os itens estão em contato.

Figura 17 – Função ϕ entre dois retângulos.

Fonte: [Stoyan et al. \(2001\)](#).

Devido à falta de algoritmos para gerar as funções de maneira prática quaisquer que sejam os itens, [Stoyan et al. \(2001\)](#) propuseram como alternativa representá-los por formas primárias (círculos¹, retângulos, polígonos convexos e polígonos regulares), e derivou funções

¹ Em geral, o que vemos na literatura é uma aproximação de itens circulares por polígonos. No entanto, este

ϕ para combinações delas. Stoyan *et al.* (2004) estenderam o trabalho anterior ao introduzir funções ϕ para outros tipos de formas, chamadas de complexas, que são aquelas que apresentam, por exemplo, curvas e/ou concavidades. Neste caso, a ideia é considerar as formas complexas como sendo combinações (união ou intersecção) de formas primárias, e a partir de tal assunção, portanto, derivar as funções.

3.2.4 Nofit polygon

O termo *nofit polygon* foi utilizado pela primeira vez por Albano e Sapuppo (1980), mas seu conceito já havia sido introduzido por Art (1966), o qual, alternativamente, utilizou o termo *shape envelope*. O *nofit polygon* nada mais é do que um polígono derivado a partir de outros dois polígonos.

Sejam A e B dois polígonos, em que A possui posição fixa. O polígono resultante da operação de deslizar B (polígono orbital) ao longo da fronteira de A (polígono fixo) é chamado de *nofit polygon* entre A e B e é denotado por NFP_{AB} . Esta operação de deslizamento ocorre de modo que B sempre toca A , mas nunca o intercepta. Note que a fronteira do NFP_{AB} é obtida a partir de um ponto de referência R de B conforme este desliza ao redor de A (Figura 18).

Figura 18 – *nofit polygon* entre os polígonos A e B .

Fonte: Elaborada pelo autor.

Assim, a partir da própria definição do *nofit polygon*, segue-se imediatamente que:

- se o ponto de referência de B está no interior do NFP_{AB} , então A e B se sobrepõem (Figura 19(a));
- se o ponto de referência de B está na fronteira do NFP_{AB} , então A e B estão em contato, mas não sobrepostos (Figura 19(b));

método, em particular, permite manter a forma original de itens circulares. É devido a este fato que não particularizamos o método para polígonos.

- se o ponto de referência de B está no exterior do NFP_{AB} , então A e B não se sobrepõem (Figura 19(c)).

Figura 19 – (a) A e B sobrepostos, (b) A e B se tocando e (c) A e B separados.

Fonte: Elaborada pelo autor.

Logo, podemos concluir que, para determinar se dois polígonos A e B estão sobrepostos, basta calcularmos o NFP_{AB} e verificarmos se o ponto de referência R do polígono orbital B está no interior do NFP_{AB} . Nos casos em que ambos os polígonos são convexos e, portanto, o *nofit polygon* também, para realizar tal verificação, podemos utilizar a função D como descrito a seguir. Sejam $a_1, a_2, \dots, a_n$ as arestas do NFP_{AB} e suponhamos que todas estejam orientadas no sentido anti-horário. Assim,

- se $D_{a_i R} > 0$ para todo $i \in \{1, 2, \dots, n\}$, então R está no lado esquerdo de cada aresta a_i e, portanto, pertence ao interior do NFP_{AB} . Logo, A e B estão sobrepostos;
- se $D_{a_i R} = 0$ para algum $i \in \{1, 2, \dots, n\}$, então R pertence à reta suporte da aresta a_i e, portanto, possivelmente à a_i . Logo, A e B estão no máximo se tocando;
- se $D_{a_i R} < 0$ para algum $i \in \{1, 2, \dots, n\}$, então R está no lado direito de a_i e, portanto, fora da região do NFP_{AB} . Logo, A e B estão separados.

Cunninghame-Green (1989) apresentou um algoritmo simples e rápido para calcular o *nofit polygon* entre dois polígonos convexos quaisquer, que consiste basicamente dos seguintes passos:

1. Considere que A (polígono fixo) possui orientação anti-horária e B (polígono orbital) possui orientação horária (Figura 20(a));
2. Translade cada uma das arestas orientadas de A e B para um ponto qualquer, como mostra a Figura 20(b);
3. Para obter o NFP_{AB} , concatene as arestas transladadas previamente adotando como critério a ordem não-decrescente de ângulos (Figura 20(c)).

Figura 20 – (a) Orientações de A e B ; (b) arestas de A e B transladadas e (c) NFP_{AB} .

Fonte: Elaborada pelo autor.

Quando há o envolvimento de polígonos não-convexos, tanto o algoritmo acima, como a técnica que utiliza a função D para identificar a posição relativa entre o ponto de referência do polígono orbital e o *nofit polygon*, não são aplicáveis. Neste caso, supondo que A e B sejam ambos não-convexos, uma alternativa, também proposta por [Cuninghame-Green \(1989\)](#), seria decompô-los e representá-los por conjuntos de polígonos convexos, e calcular o *nofit polygon* de cada polígono convexo de A , em sua respectiva posição em A , com cada polígono convexo de B . Deste modo, conseguimos utilizar a função D para verificar se há sobreposição entre pelo menos um polígono convexo de A e um polígono convexo de B .

Existem, na literatura, outras maneiras de calcular o *nofit polygon* entre dois polígonos. A soma de Minkowski, por exemplo, constitui uma alternativa bastante interessante, porém um pouco mais complexa do que o que vimos aqui. Para mais detalhes, ver [Bennell e Oliveira \(2008\)](#). Por fim, é importante mencionar que calcular o *nofit polygon* é computacionalmente custoso, mas isso pode e geralmente é feito em uma etapa de pré-processamento. Ou seja, calcula-se previamente o *nofit polygon* entre todos os pares de itens do problema e guardam-se estas informações para que sejam, mais tarde, utilizadas pelos métodos de resolução.

3.3 Posicionamento de itens no interior de um recipiente

Na seção anterior, vimos quatro diferentes métodos para verificar a sobreposição entre itens. Mas, além disso, também devemos nos preocupar com o fato de que os mesmos devem estar inteiramente contidos no recipiente. Quando optamos pela abordagem com malhas, para que isto ocorra, basta alocar os itens de modo que todas as posições de suas malhas cubram um conjunto de posições da malha do recipiente. Em relação à representação poligonal, existe um método bastante utilizado para definir uma região no recipiente na qual um item pode ser legalmente empacotado, chamado de *innerfit polygon* ([Bennell; Oliveira, 2009](#)), cuja ideia é derivada do *nofit polygon* e está descrita a seguir.

Considere um item I e um recipiente R de modo que I esteja inteiramente contido em

R e em contato com a sua fronteira. O *innerfit polygon* entre R e I , denotado por IFP_{RI} , é a região definida a partir de um ponto de referência S do item I conforme este desliza ao longo da fronteira do recipiente R . Esta operação de deslizamento deve ocorrer de modo que o item sempre toque a fronteira de R , mas nunca a intercepte. A Figura 21 ilustra um exemplo cujo recipiente é, em particular, retangular. Neste caso, o *innerfit polygon* pode, alternativamente, ser chamado de *innerfit rectangle*.

Figura 21 – *Innerfit polygon* entre um recipiente R e um item I .

Fonte: Elaborada pelo autor.

Assim, segue-se imediatamente da definição de *innerfit polygon* que:

- se o ponto de referência S do item I está no interior do IFP_{RI} , então I está inteiramente contido no recipiente R (Figura 22(a));
- se o ponto de referência S do item I está na fronteira do IFP_{RI} , então I está inteiramente contido no recipiente R e em contato com sua fronteira (Figura 22(b));
- se o ponto de referência S do item I está no exterior do IFP_{RI} , então I não está inteiramente contido no recipiente R (Figura 22(c)).

Figura 22 – Três diferentes situações representadas pelo *innerfit polygon*.

Fonte: Elaborada pelo autor.

Observe, na Figura 21, que o IFP_{RI} poderia ser simplesmente calculado subtraindo-se o comprimento e a largura do recipiente do comprimento e largura do item, respectivamente, uma vez que R é retangular. Deste modo, para determinar a localização do IFP_{RI} no recipiente, basta

posicionar o item I no canto inferior esquerdo do recipiente, de modo que ele esteja em contato com a fronteira (sem sobreposição) e, então, posicionar o vértice inferior esquerdo do IFP_{RI} no ponto de referência de I . Este procedimento todo, no entanto, não é trivial quando o recipiente é irregular. Uma alternativa para este caso está descrita no parágrafo a seguir.

Sejam J um item e T um recipiente irregular (Figura 23(a)). Primeiro, como já visto na Seção 3.1, representamos T por meio de um recipiente retangular com um conjunto de n polígonos convexos, denotado aqui por T' . Para tornar o procedimento mais fácil de ser entendido, denotamos por P_i , $i = 1, \dots, n$, cada um dos polígonos convexos de T' (Figura 23(b)). Em seguida, calculamos o *innerfit polygon* entre o retângulo T' e o item J , $IFP_{T'J}$ (Figura 23(c)), e o *nofit polygon* entre cada P_i (fixo) e o item J (orbital), NFP_{P_iJ} (Figura 23(d)). Deste modo, a região do recipiente irregular T onde o item J pode ser posicionado legalmente é dada pelo $IFP_{T'J}$ menos as regiões de intersecção entre o $IFP_{T'J}$ e cada NFP_{P_iJ} em sua respectiva posição no recipiente (Figura 23(e)).

Figura 23 – (a) Um item J e um recipiente irregular T ; (b) Representação de T por um retângulo com polígonos convexos, T' ; (c) $IFP_{T'J}$; (d) NFP_{P_1J} e NFP_{P_2J} e (e) IFP_{TJ} .

Fonte: Elaborada pelo autor.

Para calcular o *innerfit polygon* entre um recipiente irregular e um item, podemos também utilizar um método conhecido como diferença de Minkowski, cujos detalhes podem ser encontrados em Fischetti e Luzzi (2009).

3.4 Métodos de resolução

Esta seção tem por objetivo discorrer sobre métodos de resolução da literatura para o problema de corte de itens irregulares em faixa, já que constitui um dos problemas mais estudados na área, e para problemas de corte de itens irregulares em recipientes fechados, uma vez que o problema investigado neste trabalho pertence a esta classe. No problema de corte de itens irregulares em faixa, o recipiente é retangular com largura definida, e o objetivo é empacotar uma demanda de itens de modo que o comprimento do recipiente, considerado “infinito”, seja mínimo. Com relação aos problemas de corte de itens irregulares em recipientes fechados, isto é, recipientes de dimensões fixas, os objetivos podem ser variados: maximizar a ocupação de espaços de um único recipiente ou múltiplos recipientes, minimizar a quantidade de recipientes utilizados para o empacotamento, etc.

Segundo [Fowler, Paterson e Tanimoto \(1981\)](#), os problemas de corte de itens irregulares são *NP*-completo. Devido a isso, acaba sendo predominante o uso de heurísticas e meta-heurísticas para resolvê-los, as quais têm por objetivo encontrar uma solução factível de boa qualidade em um tempo computacional razoável. De fato, são poucos os trabalhos da literatura que modelam estes problemas e os resolvem utilizando métodos exatos. Na primeira parte desta seção, focamos em trabalhos que modelam os problemas de corte de itens irregulares em faixa e em recipientes fechados e, quando possível, os resolvem de forma exata. Na segunda parte, direcionamos as discussões às heurísticas.

3.4.1 Modelos e métodos exatos

3.4.1.1 Corte de itens irregulares em faixa

A seguir, apresentamos uma breve descrição a respeito de cada um dos trabalhos encontrados na literatura que modelam e resolvem o problema de corte de itens irregulares em faixa.

[Ribeiro, Carravilla e Oliveira \(1999\)](#) utilizaram programação por restrições (*constraint logic programming*) para abordar o problema, no qual consideraram apenas itens convexos. Segundo os autores, a programação por restrições é vantajosa para esta classe de problemas porque não obriga a utilização de um modelo rígido. Algumas características de um problema que às vezes são difíceis de serem encaixadas em um modelo matemático, podem ser facilmente integradas na programação por restrições. Para restringir a sobreposição entre itens e garantir que cada um deles fique inteiramente contido no recipiente, foram utilizados os métodos do *nofit polygon* e *innerfit polygon*, respectivamente. Com relação ao método de resolução, foi utilizado o solver *CLP-FD of Sicstus Prolog* ([CARLSSON; OTTOSSON; CARLSON, 1997](#)). Em 2003, os autores estenderam o trabalho ao incluir itens não-convexos ([CARRAVILLA; RIBEIRO; OLIVEIRA, 2003](#)).

Fischetti e Luzzi (2009) introduziram o primeiro modelo linear inteiro misto para o problema de corte de itens irregulares em faixa. Os autores optaram por representar os objetos do problema por polígonos e, para definir o posicionamento do ponto de referência de cada item no interior do recipiente, utilizaram variáveis contínuas. Além disso, dado o conjunto P de itens do problema, para cada par i e j de P , uma restrição de não-sobreposição envolvendo variáveis binárias foi gerada utilizando o conceito de fatias, que são partições do complemento do NFP_{ij} . Neste caso, para que os itens i e j não se sobreponham, o ponto de referência de j deve ser posicionado em qualquer uma das fatias do complemento do NFP_{ij} . Vale mencionar que os autores não especificaram técnica alguma para definir as fatias, e que, para resolver o problema, utilizaram um algoritmo *Branch & Bound* com uma estratégia de *branching* desenvolvida por eles.

Alvarez-Valdes, Martinez e Tamarit (2013) introduziram duas formulações lineares inteiras mistas, $HS1$ e $HS2$, baseadas em Fischetti e Luzzi (2009). Com relação à $HS1$, os autores propuseram dividir a região complementar do NFP_{ij} em fatias horizontais. Quando o NFP_{ij} é não-convexo, antes de definir as fatias, polígonos são alocados nas regiões côncavas com o objetivo de transformá-lo em um polígono convexo. Quanto à formulação $HS2$, esta foi obtida aplicando-se uma técnica conhecida como *lifting* nas restrições de $HS1$ que asseguram que cada item fique inteiramente contido no interior recipiente. Visando desenvolver um algoritmo *Branch & Bound* que fosse o mais eficiente possível, os autores cuidadosamente exploraram diferentes estratégias de *branching* e diferentes procedimentos para determinar limitantes inferiores para a solução e para fixar variáveis. De acordo com os experimentos, a formulação $HS2$ e a estratégia de *branching* proposta por Fischetti e Luzzi (2009) foram as que apresentaram maior eficiência.

Toledo et al. (2013) desenvolveram um modelo linear inteiro misto (*Dotted-board model*), no qual os itens são representados por polígonos e o recipiente por uma malha. Nesta formulação, cada variável de decisão δ_t^d está associada a um ponto d da malha, $d \in \{1, 2, \dots, D\}$, e a um tipo de item t , $t \in \{1, 2, \dots, T\}$, em que D e T são o número total de pontos da malha e de tipos de itens, respectivamente. Deste modo, $\delta_t^d = 1$, se o ponto de referência do item do tipo t está posicionado no ponto d , e $\delta_t^d = 0$, caso contrário. Vale mencionar que estas variáveis são definidas somente para os pontos d que pertencem ao *innerfit polygon* do item t , uma vez que os itens devem estar inteiramente contidos no recipiente. Para evitar a sobreposição entre os itens, os autores utilizaram o *nofit polygon*. O modelo, que foi resolvido utilizando o *solver* de programação inteira mista CPLEX 12.3, é bastante interessante para problemas que possuem poucos tipos de itens, uma vez que o número total de variáveis binárias depende apenas da quantidade de tipos de itens e não do número total dos mesmos.

Leao et al. (2016) introduziram um modelo linear inteiro misto para o problema em questão cujo recipiente é semi-contínuo, no qual o eixo x é mantido contínuo enquanto que o eixo y é discretizado. Em outras palavras, o recipiente é formado por um conjunto de listras horizontais, e o ponto de referência de cada item só pode ser posicionado sobre elas. Para evitar

a sobreposição entre itens e garantir que eles fiquem inteiramente contidos no recipiente, foram escritas restrições baseadas nos métodos do *nofit polygon* e *innerfit polygon*, respectivamente. O modelo, que foi resolvido utilizando o CPLEX 12.5, apresenta vantagens sobre o modelo discreto de Toledo *et al.* (2013), uma vez que, diferentemente deste último, mostrou-se capaz de resolver instâncias com recipientes grandes. Por outro lado, quando comparado ao modelo contínuo HS2 de Alvarez-Valdes, Martinez e Tamarit (2013), o modelo semi-contínuo, no geral, obteve menor desempenho.

Cherri *et al.* (2016) apresentaram dois modelos para o problema de corte de itens irregulares em faixa, chamados de *Direct Trigonometry Model (DTM)* e *NFP Covering Model (NFP-CM)*. Assim como os demais, estes também são lineares inteiros mistos e, até onde sabemos, são os últimos modelos da literatura para este problema. As restrições de não-sobreposição do modelo DTM são baseadas no método da trigonometria direta, enquanto que o modelo NFP-CM tem essas mesmas restrições baseadas no *nofit polygon*, como sugerem os próprios nomes. Segundo os autores, ambos os modelos são bastante robustos uma vez que podem ser utilizados para qualquer tipo de item não-convexo, com ou sem buracos e, além disso, são mais fáceis de serem implementados quando comparados aos modelos anteriores. Experimentos computacionais mostraram que o modelo NFP-CM supera não apenas o DTM, mas também o HS2 de Fischetti e Luzzi (2009), tido até então como o melhor modelo da literatura.

3.4.1.2 Corte de itens irregulares em recipientes fechados

Com relação aos problemas de corte de itens irregulares em recipientes fechados, encontramos na literatura dois trabalhos que trazem modelos, mas não os resolvem de fato. A seguir, apresentamos alguns detalhes acerca de cada um deles.

Scheithauer e Terno (1993) introduziram modelos lineares inteiros mistos para o problema da mochila envolvendo um único recipiente. Neste caso, dado um conjunto de itens com diferentes valorações, o objetivo é determinar um subconjunto de itens cujo empacotamento no recipiente seja factível e a valoração seja máxima. A princípio, os autores modelaram o problema para objetos retangulares e, em seguida, para objetos irregulares, sendo estes representados por polígonos em ambas as situações. Neste último caso, foram propostos dois diferentes modelos, sendo um deles exclusivamente para objetos representados por polígonos convexos e outro para objetos representados por polígonos arbitrários. Para escrever as restrições de não-sobreposição e de contenção dos itens no recipiente, foram utilizados os métodos do *nofit polygon* e *innerfit polygon*, respectivamente, chamados neste trabalho de *outer hodograph* e *inner hodograph*.

Baldacci *et al.* (2014) também modelaram o problema da mochila com um único recipiente, porém considerando exclusivamente objetos irregulares, sendo que o recipiente, além de irregular, possui defeitos e zonas de qualidades. Vale ressaltar que os autores optaram pela utilização de malhas, como já discutido na Seção 3.1. Devido ao seu grande número de variáveis, o modelo linear inteiro, chamado de SNP, não pode ser diretamente utilizado para resolver o

problema. Sendo assim, os autores derivaram uma relaxação lagrangeana a partir de *SNP* e resolveram seu dual, o qual, infelizmente, forneceu um limitante superior muito fraco. Apesar disto, [Baldacci et al. \(2014\)](#) consideraram a relaxação interessante e inspiraram-se nela para desenvolver métodos heurísticos. Além das formulações citadas, também foi introduzido o modelo linear inteiro *MNP* para o problema de corte itens irregulares em múltiplos recipientes, cujo objetivo é atender a uma demanda de itens minimizando o custo total dos recipientes utilizados. Este modelo, assim como os demais, também não foi diretamente resolvido.

3.4.2 Métodos heurísticos

Nesta seção, estendemos o estudo de métodos de resolução para os problemas abordados nesta seção, trazendo agora métodos heurísticos. Contudo, antes, fazemos uma abordagem conceitual sobre heurísticas construtivas e heurísticas de melhoramento. As informações aqui contidas estão fortemente baseadas em [Bennell e Oliveira \(2009\)](#).

Heurísticas construtivas buscam construir uma solução de forma parcial, selecionando o item mais apropriado para ser alocado no recipiente a cada iteração. A qualidade das soluções obtidas por estas heurísticas depende principalmente da ordem em que os itens são alocados no recipiente, que pode ser definida de forma aleatória, ou considerando o tamanho dos itens, por exemplo, e da regra de posicionamento escolhida, cujo objetivo é buscar posições factíveis no recipiente para cada item. Utilizada em problemas de corte de itens irregulares pela primeira vez por [Art \(1966\)](#), a heurística *bottom-left (BL)* tem sido bastante escolhida pelos pesquisadores como regra de posicionamento, uma vez que é de fácil entendimento e produz boas soluções em baixo tempo computacional. Seu objetivo é alocar cada item, iterativamente, na posição factível mais à esquerda e abaixo possível no recipiente, como mostra o exemplo da Figura 24.

Figura 24 – Exemplo de aplicação da regra *BL*.

Fonte: [Bennell e Oliveira \(2009\)](#).

Heurísticas de melhoramento, por sua vez, visam aperfeiçoar uma solução completa já existente através de modificações nesta solução. Quando uma solução é representada através de uma sequência de itens, as duas principais estratégias para modificá-la são a inserção e a permutação, que consistem, respectivamente, em trocar dois itens de posição na sequência ou apenas inserir um item em uma outra posição. Para avaliar a qualidade de uma sequência

modificada, é necessário empacotar os itens desta nova sequência no recipiente e analisar o leiaute obtido. Estas mesmas estratégias podem ser utilizadas quando opta-se por representar a solução diretamente através de um leiaute. No caso da inserção, um único item é movido para uma outra posição no recipiente, enquanto que na permutação, dois ou mais itens têm suas posições trocadas. Alguns pesquisadores têm optado, ainda, por outro mecanismo, chamado de compactação e separação, no qual todos os itens do problema são movidos de posição simultaneamente.

3.4.2.1 Problemas de corte de itens irregulares em faixa

Existe uma grande variedade de trabalhos na literatura que resolvem o problema de corte de itens irregulares em faixa utilizando métodos heurísticos. Devido a este fato, selecionamos alguns trabalhos interessantes e que possuem diferentes metodologias entre si para discurtirmos nos parágrafos a seguir.

[Dowsland, Dowsland e Bennell \(1998\)](#) desenvolveram uma heurística de melhoramento (*the jostle approach*) que oscila entre empacotar os itens no lado esquerdo e direito do recipiente. Segundo os autores, esta técnica de “jogar” os itens de um lado para o outro do recipiente, quando comparada com heurísticas que os alocam apenas em um lado, produz leiautes mais uniformes com relação às bordas laterais. Os autores optaram por representar o recipiente por uma malha, reduzindo assim para finito o conjunto de posições, e utilizaram os métodos da trigonometria direta e *nofit polygon* para evitar a sobreposição entre os itens, o que acabou gerando, portanto, duas diferentes implementações, uma para cada método. Resultados mostraram que a heurística é interessante de ser utilizada quando o tempo computacional disponível é estritamente limitado, porém maior do que o tempo exigido por algum método de alocação de itens de passo único.

[Oliveira, Gomes e Ferreira \(2000\)](#) introduziram uma heurística construtiva chamada TOPOS (*Técnicas de Optimização para o Posicionamento de Figuras Irregulares*), na qual a solução é construída parcialmente de modo que, em cada iteração, o conjunto de itens previamente alocados no recipiente é considerado como um único item, cuja representação é dada pelo polígono obtido a partir do contorno externo do grupo de itens já posicionados. No geral, a ideia do algoritmo é, em cada iteração, determinar qual o melhor item para ser alocado (entre aqueles ainda restantes), sua rotação, e qual o melhor ponto para posicioná-lo. Os pontos candidatos para a alocação de um item são obtidos a partir das arestas do seu *nofit polygon* com o item que representa a solução parcial. Esta estratégia faz com que os itens se encostem, tornando o leiaute mais compacto. A partir dos experimentos computacionais, os autores definiram o TOPOS como uma boa alternativa para outros métodos de resolução existentes na literatura.

[Gomes e Oliveira \(2002\)](#) apresentaram uma heurística de melhoramento (*the 2-exchange nesting heuristic*) que realiza busca local sobre o espaço de soluções consistindo de sequências de itens. Neste método, soluções vizinhas são obtidas permutando-se pares de itens em uma sequência, e o tamanho da vizinhança é controlado por um parâmetro Δ que limita as modificações

a uma parte da sequência. Assim, cada item que está dentro da distância estabelecida na sequência é permutado com cada um dos itens seguintes que também estão dentro desta distância. Em cada iteração, o algoritmo move da solução que está sendo modificada (solução atual), para uma solução vizinha melhor, isto é, uma solução que produza um leiaute de menor comprimento. Os experimentos computacionais, que foram realizados com cinco instâncias da literatura, mostraram que o método foi capaz de melhorar os resultados de quatro delas.

Gomes e Oliveira (2006) introduziram um algoritmo híbrido de melhoramento que consiste de um algoritmo de recozimento simulado (*simulated annealing*) responsável por guiar a busca sobre o espaço de soluções, neste caso, leiautes, juntamente com dois modelos de programação linear (um de separação e outro de compactação) utilizados para gerar soluções vizinhas. Diferentemente de algoritmos de busca local pura, algoritmos de recozimento simulado permitem alguns movimentos controlados de modo a tentar atingir otimalidade global. A estratégia adotada neste trabalho para gerar soluções vizinhas consiste de três etapas: (i) modificar o leiaute movendo-se itens de posição; (ii) utilizar o modelo de separação para remover qualquer sobreposição que tenha eventualmente ocorrido; (iii) compactar o leiaute através do modelo de compactação. Experimentos computacionais provaram a eficiência do método, uma vez que os resultados de 15 instâncias da literatura foram melhorados.

Elkeran (2013) desenvolveu um método bastante eficiente para o problema de corte de itens irregulares em faixa, considerado atualmente como o melhor da literatura, uma vez que melhorou o resultado de várias instâncias clássicas. O método, chamado de *guided cuckoo search* (GCS), é uma combinação da técnica *cuckoo search*, de Yang e Deb (2009), com técnicas de otimização de busca local guiada. O algoritmo consiste, iterativamente, de duas fases: a primeira, é responsável por minimizar o comprimento do recipiente, enquanto que a segunda fase resolve sobreposições que ocorrem durante a execução do algoritmo. É interessante mencionar que, para diminuir a complexidade do problema, os autores desenvolveram uma estratégia bastante interessante, chamada de *pairwise clustering*, cuja função é, em uma fase de pré-processamento, agrupar os itens em pares de acordo com características que se combinam.

3.4.2.2 Problemas de corte de itens irregulares em recipientes fechados

Com relação ao corte de itens irregulares em recipientes fechados, buscamos, na literatura, por trabalhos que lidam especificamente com recipientes irregulares contendo defeitos e/ou zonas de qualidades, uma vez que este foi o foco desta pesquisa. É interessante mencionar que na maioria destes trabalhos foram realizados testes computacionais utilizando instâncias reais obtidas em indústrias que utilizam o couro como matéria-prima.

Heistermann e Lengauer (1995) resolveram o problema de corte de itens irregulares com o objetivo de maximizar a ocupação de um único recipiente irregular com defeitos e zonas de qualidades. Até onde sabemos, este é o primeiro trabalho da literatura que apresenta resultados com instâncias reais de indústrias que utilizam o couro. Os objetos do problema foram

representados por polígonos e a heurística desenvolvida realiza, iterativamente, as seguintes etapas: (i) determina a região do recipiente mais promissora para posicionar o próximo item; (ii) seleciona um conjunto de itens candidatos a próximo item; (iii) aloca todos os itens do conjunto anterior, provisoriamente, e mantém alocado aquele que fornece o melhor leiaute; (iv) caso o passo anterior tenha gerado posicionamentos ilegais para todos os itens, algumas medidas são tomadas para evitar que o algoritmo entre em um laço sem fim. A execução termina quando todos os itens são alocados ou quando não há mais espaço suficiente no recipiente. Os testes mostraram que o método é eficiente, com um baixo tempo computacional.

Crispin *et al.* (2005) introduziram dois algoritmos genéticos para o resolver um problema real cujos itens, que são partes que compõem calçados, devem ser cortados a partir do couro. Neste caso, o objetivo é maximizar a ocupação de um único recipiente. Para cada algoritmo, foi proposta uma diferente metodologia. No primeiro, ao inserir um novo item, um polígono chamado de *window region*, é posicionado no centro do último item inserido e o algoritmo busca, dentro desta região, por um ponto cujo posicionamento do novo item maximize a utilização local de espaços. O ponto, neste caso, é obtido a partir das arestas do *nofit polygon* do novo item com o item posicionado anteriormente. Quanto ao segundo algoritmo, a ideia é inserir os itens sequencialmente de modo que se conectem a determinados itens já alocados no recipiente. O ponto para alocar um item é obtido a partir da intersecção dos *nofit polygons* dele com os itens nos quais pode encostar. Com relação aos experimentos, o primeiro algoritmo produziu melhores resultados para as intâncias testadas.

Lee, Ma e Cheng (2008) apresentaram uma heurística chamada de *quick location and movement (QLM)* para resolver o problema de corte de itens irregulares em múltiplos recipientes irregulares com defeitos. O método consiste basicamente de duas grandes etapas: na primeira, os itens são transformados em polígonos através de uma técnica introduzida pelos próprios autores; na segunda etapa, os mesmos são ordenados por ordem decrescente de área, e então inseridos um a um nos recipientes. Para alocar um item, após determinar uma posição inicial considerada boa, o algoritmo move e rotaciona o item até encontrar a melhor posição final. Quando o algoritmo não encontra uma posição factível para alocar um item em um determinado recipiente, ele deixa o usuário optar entre tirar alguns itens deste recipiente ou escolher outro, e então tenta alocá-lo novamente. A partir dos experimentos computacionais, os autores concluíram que o *QLM* é bastante eficiente, inclusive quando aplicado a itens e recipientes com formas bastante irregulares.

Zhang e Yang (2009) desenvolveram um método inovador, chamado de *Simulated Annealing Based Genetic Algorithm (SABGA)*, para o resolver o problema de corte de itens irregulares em couro. Vale enfatizar que os autores consideraram múltiplos recipientes, sendo que estes possuem apenas defeitos, e que optaram por representar os objetos do problema por malhas. O método, que possui este nome pelo fato de que utiliza estratégias de algoritmos genéticos e de recozimento simulado, é responsável por controlar a ordem na qual os itens são inseridos nos

recipientes, bem como suas rotações. Os pontos de posicionamento, neste caso, são selecionados pela heurística *bottom-left*. De acordo com os experimentos computacionais, o método é capaz de empacotar diferentes tipos de itens em múltiplos diferentes recipientes de forma eficaz e com um tempo de execução considerado aceitável pelas indústrias. É interessante mencionar que em momento algum os autores mencionaram o uso de instâncias reais.

Alves *et al.* (2012a) resolveram um problema real da indústria automotiva, no qual os itens, que são partes de assentos de carros, devem ser cortados a partir de peças de couro com defeitos e zonas de qualidades. Neste caso, o objetivo é empacotar um conjunto de itens utilizando o menor número possível de recipientes. Os autores optaram por representar os objetos do problema por polígonos, e utilizaram o método do *nofit polygon* para evitar a sobreposição entre os itens. Até onde sabemos, esta é a primeira vez em que o *nofit polygon* é utilizado na resolução do problema de corte de itens irregulares em couro. Para garantir que os itens fiquem inteiramente contidos no recipiente, foi utilizado o método do *innerfit polygon*. Quanto aos métodos de resolução, os autores desenvolveram diferentes estratégias para cada etapa que compõe uma heurística construtiva e a partir da combinação destas estratégias, definiram várias heurísticas construtivas. Os experimentos computacionais, que foram realizados utilizando instâncias reais, mostraram que os métodos podem gerar leiautes de alta qualidade.

Alves *et al.* (2012b) resolveram o mesmo problema do trabalho anterior, mas agora com o objetivo de maximizar a ocupação de um único recipiente. O método de resolução desenvolvido consiste de uma meta-heurística de busca local baseada em busca em vizinhança variável. Até onde sabemos, este é o primeiro método de busca local da literatura para o problema de corte em couro. Neste caso, a busca é feita sobre a sequência de itens, e soluções vizinhas são obtidas a partir de quatro diferentes movimentos na sequência. Tanto para obter uma solução inicial, como para gerar o leiaute após cada movimento na sequência de modo a avaliar sua qualidade, os autores fizeram uso de uma das heurísticas construtivas apresentadas em Alves *et al.* (2012a). A execução do método chega ao fim quando o limite de tempo fornecido como entrada é alcançado. De acordo com os experimentos computacionais, a meta-heurística foi capaz de melhorar as soluções iniciais providas a partir da heurística construtiva, e forneceu soluções cuja qualidade compete com aquelas obtidos por humanos.

Baldacci *et al.* (2014) investigaram o problema de corte de itens irregulares em recipientes com defeitos e zonas de diferentes qualidades considerando, a princípio, um único recipiente irregular e, em seguida, mais de um recipiente, como já discutido na Seção 3.1. Neste trabalho, os objetos do problema foram representados por malhas, uma vez que, segundo os autores, esta representação suporta um processamento mais eficiente das regiões defeituosas. Foram desenvolvidas três heurísticas para resolver o problema considerando um único recipiente, chamadas de *SNH1*, *SNH2*, e *SNH3*, as quais se diferem quanto à regra de alocação dos itens, e uma heurística para o problema com múltiplos recipientes, chamada de *MCH*. Para validar as heurísticas desenvolvidas, foram utilizadas tanto instâncias da literatura, como instâncias reais

obtidas em indústrias que têm o couro como matéria-prima. A partir dos experimentos, os autores concluíram que os métodos podem resolver problemas reais em um baixo tempo computacional e são capazes de produzir soluções de boa qualidade.

Pinto *et al.* (2016) continuaram a investigar o problema da indústria automotiva visto em Alves *et al.* (2012a) e Alves *et al.* (2012b). Neste caso, foram desenvolvidas uma heurística construtiva e um algoritmo de busca local para resolver o problema. De forma resumida, a heurística construtiva, em cada iteração, simula o posicionamento de alguns itens em determinados pontos, e seleciona, para de fato ser inserido, aquele cujo posicionamento maximiza o valor da função que avalia a qualidade dos leiautes. Com relação ao algoritmo de busca local, para gerar uma solução vizinha, o procedimento consiste em simular a remoção de um item do atual leiaute e, no espaço vazio obtido, aplicar uma heurística construtiva similar à anterior, porém agora inserindo itens ainda não alocados. Esta mesma heurística construtiva é utilizada para gerar uma solução inicial. Os experimentos computacionais mostraram que, quando comparado com a heurística construtiva, o algoritmo de busca local possui maior potencial.

MÉTODOS DE RESOLUÇÃO PROPOSTOS

Para a resolução do problema bidimensional de corte de itens irregulares em recipientes irregulares com defeitos e zonas de qualidades, abordado neste trabalho (ver Capítulo 2), desenvolvemos várias heurísticas construtivas, as quais diferem-se umas das outras em relação a três diferentes estratégias: ordenação dos itens, ordenação dos recipientes e busca de posicionamento de itens. Sendo assim, escrevemos um algoritmo básico, isto é, um algoritmo mais geral, a partir do qual variações podem ser obtidas fazendo-se adaptações em algumas de suas linhas de acordo com as estratégias selecionadas. Na Seção 4.1, descrevemos separadamente os principais componentes do algoritmo básico e, em seguida, apresentamos-no de fato. Na Seção 4.2, por fim, definimos as heurísticas propostas neste trabalho após a introdução das estratégias mencionadas.

4.1 Heurística construtiva básica

Antes de entrarmos em mais detalhes a respeito dos métodos, é importante trazer algumas informações sobre o modo como representamos os objetos do problema. Com relação aos itens, optamos por representar cada um deles por meio de um conjunto de polígonos convexos (Seção 3.1). Cada recipiente irregular, por sua vez, é representado através de seu menor retângulo envolvente com um conjunto de defeitos posicionados em pontos estratégicos (Seção 3.1). Assim como os itens, também representamos os defeitos e zonas de qualidades dos recipientes por meio de conjuntos de polígonos convexos. Por fim, observe que podemos tratar os defeitos como se fossem zonas de pior qualidade, sobre as quais nenhum item pode ser posicionado, não sendo necessário, portanto, fazer diferenciação entre os termos. Na Seção 4.3, retomamos estas informações de forma mais detalhada.

Restrições de não-sobreposição e contenção

Para que o posicionamento de um item em um determinado ponto (x,y) do recipiente

seja factível¹, em que x e y são coordenadas reais, sabemos que o mesmo não deve sobrepor qualquer item já alocado, qualquer zona de qualidade indesejável² e que, além disso, deve estar inteiramente contido no recipiente. Para realizar tais verificações, utilizamos os métodos do *nofit polygon* (Seção 3.2.4) e *innerfit polygon* (Seção 3.3), respectivamente. Optamos por realizar os cálculos dos *nofit polygons* em uma fase de pré-processamento, ou seja, previamente calculamos os *nofit polygons* entre todos os pares de itens (incluindo cada item com ele mesmo), e entre cada item (orbital) e cada zona, de cada recipiente, com qualidade inferior à mínima exigida para o item em questão. Os *innerfit polygons*, por outro lado, são calculados durante a resolução do problema conforme necessário. É importante enfatizar que, para cada item, realizamos os cálculos mencionados considerando cada uma de suas rotações, sendo estas fornecidas previamente pelo usuário.

Pontos de posicionamento de itens

Os possíveis pontos candidatos para alocação de um item i em um recipiente k , tomando como referência o trabalho de Gomes e Oliveira (2002), são obtidos a partir das arestas dos *nofit polygons* entre i (orbital) e cada item já alocado em k , entre i (orbital) e cada zona de k com qualidade inferior à qualidade mínima exigida para i , e, também, a partir das arestas do *innerfit polygon* entre i e o recipiente k . De modo a reduzir para finito este conjunto de pontos, calculamos apenas os pontos de interseção entre cada par de arestas acima. Observe que, posicionando i em algum destes pontos, a partir das próprias definições de *nofit polygon* e *innerfit polygon*, fazemos com que o mesmo encoste nas bordas do recipiente k , em zonas com qualidade inferior à mínima exigida para i , ou em itens já alocados. A Figura 25 ilustra um exemplo simples, no qual calculamos todos os possíveis pontos candidatos para a alocação de um item e circulamos apenas aqueles que geram posicionamentos factíveis.

Após determinar os pontos candidatos para alocação de um item, precisamos buscar um entre eles para de fato inseri-lo. Como já mencionado, esta é uma das etapas que diferencia uma heurística de outra. Cada uma delas, utilizando uma diferente estratégia, sobre as quais discutiremos em detalhes mais à frente, sempre seleciona um entre os quatros pontos a seguir.

- *Bottom-Left (BL)*: ponto mais à esquerda e abaixo possível no recipiente (Figura 26 (a));
- *Bottom-Right (BR)*: ponto mais à direita e abaixo possível no recipiente (Figura 26 (b));
- *Top-Right (TR)*: ponto mais à direita e acima possível no recipiente (Figura 26 (c));
- *Top-Left (TL)*: ponto mais à esquerda e acima possível no recipiente (Figura 26 (d)).

¹ Posicionar um item em um ponto (x, y) do recipiente significa fixar seu ponto de referência em (x, y) .

² Lembrando que, uma zona de qualidade é considerada indesejável, se sua qualidade é inferior à mínima exigida para o item (ver Capítulo 2).

Figura 25 – Pontos candidatos para alocação de um item.

Fonte: Elaborada pelo autor.

Figura 26 – (a) Ponto *BL*; (b) Ponto *BR*; (c) Ponto *TR* e (d) Ponto *TL*.

Fonte: Elaborada pelo autor.

Rotações

É importante deixar claro que geramos um conjunto de pontos candidatos para alocação de um item para cada uma de suas rotações³. Sendo assim, supondo que queiramos inserir um item no ponto *BL*, precisamos buscar por este ponto no conjunto de pontos de cada rotação e determinar qual deles fornece o melhor posicionamento para o item. Para isso, o algoritmo, para cada rotação do item, simula o seu posicionamento no respectivo ponto *BL* e seleciona aquele que minimiza a distância entre a borda lateral esquerda do recipiente e o vértice mais à direita do item, como mostra o exemplo da Figura 27(a). Caso obtenhamos a mesma distância para mais de uma rotação, como critério de desempate, o algoritmo seleciona o posicionamento que minimiza a distância entre a borda inferior do recipiente e o vértice mais acima do item (Figura 27(b)). Por fim, se também obtivermos um empate neste último critério, então o posicionamento com rotação de menor ângulo é escolhido. Observando o fato de que, no caso do ponto *BL*, o objetivo principal é determinar uma rotação que mantenha o item o mais próximo possível da borda esquerda (*left*) do recipiente, e que, em caso de empate, olhamos para a borda inferior (*bottom*) do recipiente, podemos facilmente adaptar este procedimento para os pontos *BR*, *TR* e *TL*.

Figura 27 – (a) Determinando a melhor rotação para inserir um item no ponto *BL* e (b) Determinando a melhor rotação para inserir um item no ponto *BL* em caso de empate no critério utilizado em (a).

Fonte: Elaborada pelo autor.

Uma vez que definimos os principais componentes do algoritmo, podemos, agora, de fato introduzi-lo. Dadas uma lista de n tipos de itens irregulares, cada um com uma demanda, um conjunto de polígonos convexos que o forma e um ponto de referência, e uma lista de m recipientes irregulares com zonas de qualidades, ambas devidamente ordenadas, a ideia é, sequencialmente, inserir cada tipo de item até atender sua demanda ou até que não caibam mais em nenhum dos recipientes. Para cada item, o algoritmo percorre a lista de recipientes

³ Lembrando que as rotações que cada item pode sofrer são definidas previamente pelo usuário.

desde o começo até encontrar um posicionamento que seja factível. Uma vez encontrado este posicionamento, o algoritmo aloca o item no respectivo recipiente. Como entrada, temos os seguintes dados: um número m de recipientes; uma lista *bins* contendo m recipientes irregulares com zonas de qualidades; um número n de tipos de itens; uma lista *items* contendo n tipos de itens, sendo que, para cada tipo de item, armazenamos sua demanda, rotações permitidas e qualidade mínima exigida; e, por fim, um conjunto *NFPs* contendo todos os *nofit polygons* calculados na etapa de pré-processamento. Como saída, temos a solução S obtida que contém as posições dos pontos de referência e rotações dos itens empacotados em cada recipiente.

A seguir, apresentamos o algoritmo básico (Algoritmo 1) para as heurísticas desenvolvidas neste trabalho.

Algoritmo 1: Algoritmo Básico**Entrada:** $m, bins, n, items, NFPs$ **Saída:** S **1 Início****2** $S \leftarrow \emptyset$;**3** $items$ é ordenado;**4** $bins$ é ordenado;**5** **para** $i = 1$ a n **faça****6** **enquanto** $demandai > 0$ **faça****7** **para** $k = 1$ a m **faça****8** **para** $r = 1$ a $numRot_i$ **faça****9** Adicione no conjunto A_{kir} as arestas do IFP_{kir} ;**10** **para** cada NFP_{wir} , em que w (fixo) são as zonas com qualidade inferior à mínima exigida para o item i , e os itens já alocados no recipiente k (caso existam) **faça****11** Insira no conjunto A_{kir} as arestas do NFP_{wir} ;**12** **fim****13** **para** cada par s e t de arestas que pertencem a A_{kir} **faça****14** Calcule o ponto de interseção p de s e t e insira-o no conjunto $dots_{kir}$;**15** **fim****16** **para** cada ponto p do conjunto $dots_{kir}$ **faça****17** **se** p pertence ao exterior do IFP_{kir} e/ou p pertence ao interior de qualquer NFP_{wir} , em que w são as zonas com qualidade inferior à mínima exigida para o item i , e os itens já alocados no recipiente k (caso existam) **então****18** Apague p do conjunto $dots_{kir}$;**19** **fim****20** **fim****21** **fim****22** Busque em cada conjunto $dots_{kir}$, $r = 1, \dots, numRot_i$, pelo ponto $(x, y)_p$, $p = BL, BR, TR$ ou TL , e determine qual deles fornece o melhor posicionamento para o item i no recipiente k na respectiva rotação r ;**23** **se** existe um ponto $(x, y)_p$, $p = BL, BR, TR$ ou TL , para inserir o item i no recipiente k em uma rotação r' **então****24** S recebe o item i , alocado no recipiente k , no ponto $(x, y)_p$, $p = BL, BR, TR$ ou TL , na rotação r' ;**25** $demandai \leftarrow demandai - 1$;**26** Retorne para a linha 6;**27** **fim****28** **fim****29** **fim****30** **fim****31** **fim****32** **retorna** S ;

4.2 Variações das heurísticas

Conhecido o algoritmo básico, podemos, finalmente, definir as heurísticas propostas, as quais, como já sabemos, diferem-se umas das outras em relação à ordenação dos itens e recipientes (linhas 3 e 4 do Algoritmo 1, respectivamente), e em relação ao modo como buscamos por posicionamentos factíveis para os itens nos recipientes (linha 22 do Algoritmo 1).

Ordenação dos itens e recipientes

Com relação à ordem na qual os itens são inseridos nos recipientes, temos duas diferentes estratégias: $DecArea_i$ e $QiDecArea_i$. A primeira, bastante utilizada na literatura (BENNELL; OLIVEIRA, 2009), insere os itens, um a um, por ordem decrescente de área. A segunda, desenvolvida por nós, separa os itens em grupos de acordo com a qualidade mínima exigida para os mesmos e, dentro de cada grupo, os ordena utilizando a estratégia $DecArea_i$. Feito isso, os grupos também são ordenados, começando por aquele cujos itens possuem o menor grau de exigência mínima de qualidade. Quanto aos recipientes, fizemos uso da estratégia $DecArea_r$, a qual os ordena começando pelo recipiente de maior área complementar às zonas de qualidades até aquele de menor área, e também mantivemos a ordem na qual o usuário fornece cada recipiente, chamada de $User_r$.

Estratégias de posicionamento de itens

1. *Spiral*: Para cada recipiente, inserimos os itens alternando sequencialmente entre os pontos BL , BR , TR e TL (Figura 28). Sendo assim, sempre que o algoritmo troca de recipiente, ele tenta inserir o novo item no ponto da sequência posterior ao ponto no qual foi inserido o último item do recipiente em questão.
2. *MinDistance*: Esta estratégia simula o posicionamento dos itens em cada um dos quatro pontos (BL , BR , TR e TL), avalia, e seleciona um entre eles para de fato inseri-los. Mais especificamente, dado um item i , primeiramente, o algoritmo posiciona i no ponto BL e calcula a distância entre o vértice inferior esquerdo do recipiente e o vértice superior direito do menor retângulo que envolve o item i . Em seguida, ele realiza este mesmo procedimento para os demais pontos, isto é, BR , TR e TL , porém, em cada caso, considerando um diferente vértice do recipiente e do retângulo envolvente, obviamente, como ilustra o exemplo da Figura 29, no qual buscamos por um posicionamento para inserir um item em um recipiente com quatro itens já alocados. Calculadas as quatro distâncias, o algoritmo seleciona a menor entre elas e insere o item no respectivo ponto. No caso do exemplo da Figura 29, uma vez que temos um empate entre as distâncias obtidas nos casos BL e BR , selecionamos o ponto que aparece primeiro na sequência, sendo, neste caso, o ponto BL .

Figura 28 – (a) *Bottom-Left*; (b) *Bottom-Right*; (c) *Top-Right* e (d) *Top-Left*.

Fonte: Elaborada pelo autor.

Figura 29 – Utilização do critério $\min Distance$ para seleccionar o melhor ponto, entre os pontos BL , BR , TR e TL , para inserir um item em um recipiente.

Fonte: Elaborada pelo autor.

3. *MinArea*: Neste caso, o princípio é o mesmo que o da *minDistance*, no entanto, supondo novamente que estejamos simulando o posicionamento de um item i no ponto BL , em vez de calcular a distância entre o vértice inferior esquerdo do recipiente e o vértice superior direito do menor retângulo que envolve i , o algoritmo calcula a área do retângulo que possui como diagonal a referida distância, ou, em outras palavras, a área do retângulo que possui como vértices opostos o vértice inferior esquerdo do recipiente e o vértice superior direito do menor retângulo que envolve o item. Calculada a área para cada um dos quatro pontos, o algoritmo seleciona a menor entre elas e insere o item no respectivo ponto. No exemplo da Figura 30, novamente selecionamos o ponto BL para inserir o item.

Figura 30 – Aplicação do critério *minArea* para selecionar o melhor ponto, entre os pontos BL , BR , TR e TL , para inserir um item em um recipiente.

Fonte: Elaborada pelo autor.

4. *Random*: A estratégia *Random*, por fim, qualquer que seja o recipiente, seleciona aleatoriamente um entre os quatro pontos para inserir os itens.

Finalmente, observe que, a partir da combinação das quatro estratégias de posicionamento (*Spiral*, *MinDistance*, *MinArea* e *Random*) com as duas estratégias de ordenação dos recipientes ($User_r$ e $DecArea_r$) mais as duas estratégias de ordenação dos itens ($DecArea_i$ e $Qi_DecArea_i$), podemos definir um total de 16 diferentes heurísticas. Além disso, optamos também por combinar as estratégias de ordenação de itens e recipientes com a estratégia de posicionamento *Bottom-Left*, obtendo, portanto, mais quatro métodos. A ideia, é poder comparar as estratégias de posicionamento propostas com alguma que seja bastante utilizada na literatura, como é o caso da *Bottom-Left* (OLIVEIRA; GOMES; FERREIRA, 2000; GOMES; OLIVEIRA, 2002; MARTINS; TSUZUKI, 2010; ALVES *et al.*, 2012a). A Tabela 1 traz todas as combinações obtidas, separadas em cinco grupos.

Tabela 1 – Heurísticas geradas a partir da combinação de diferentes estratégias.

Grupo	Heurística	Posicionamento	Ordenação dos recipientes	Ordenação dos itens
<i>S</i>	S_1	<i>Spiral</i>	$User_r$	$DecArea_i$
	S_2	<i>Spiral</i>	$User_r$	$QiDecArea_i$
	S_3	<i>Spiral</i>	$DecArea_r$	$DecArea_i$
	S_4	<i>Spiral</i>	$DecArea_r$	$QiDecArea_i$
<i>MD</i>	MD_1	<i>MinDistance</i>	$User_r$	$DecArea_i$
	MD_2	<i>MinDistance</i>	$User_r$	$QiDecArea_i$
	MD_3	<i>MinDistance</i>	$DecArea_r$	$DecArea_i$
	MD_4	<i>MinDistance</i>	$DecArea_r$	$QiDecArea_i$
<i>MA</i>	MA_1	<i>MinArea</i>	$User_r$	$DecArea_i$
	MA_2	<i>MinArea</i>	$User_r$	$QiDecArea_i$
	MA_3	<i>MinArea</i>	$DecArea_r$	$DecArea_i$
	MA_4	<i>MinArea</i>	$DecArea_r$	$QiDecArea_i$
<i>BL</i>	BL_1	<i>Bottom-Left</i>	$User_r$	$DecArea_i$
	BL_2	<i>Bottom-Left</i>	$User_r$	$QiDecArea_i$
	BL_3	<i>Bottom-Left</i>	$DecArea_r$	$DecArea_i$
	BL_4	<i>Bottom-Left</i>	$DecArea_r$	$QiDecArea_i$
<i>R</i>	R_1	<i>Random</i>	$User_r$	$DecArea_i$
	R_2	<i>Random</i>	$User_r$	$QiDecArea_i$
	R_3	<i>Random</i>	$DecArea_r$	$DecArea_i$
	R_4	<i>Random</i>	$DecArea_r$	$QiDecArea_i$

4.3 Implementação

Nesta seção, primeiramente, apresentamos como os dados de entrada, necessários para resolver o problema, são fornecidos aos métodos, os quais foram implementados em linguagem de programação C/C++. Em seguida, discutiremos sobre o modo como os dados de saída, isto é, os dados referentes à solução do problema, são retornados ao usuário.

4.3.1 Dados de entrada

Os dados de entrada do problema são armazenados em quatro diferentes tipos de arquivos, sendo que cada tipo de arquivo armazena dados específicos. Ou seja, para cada tipo de item, temos um arquivo contendo suas informações, assim como para cada recipiente; temos, também, um arquivo que armazena os dados gerais do problema; e, por fim, um arquivo para cada *nofit polygon* calculado na etapa de pré-processamento. A seguir, de forma mais detalhada, definimos o modo como fornecemos todos os dados de entrada aos métodos.

Itens

Como lidamos com muitos itens não-convexos, optamos por representar cada um deles por um conjunto de polígonos convexos, como feito em [Cherri et al. \(2016\)](#). Esta abordagem nos permite escapar da dificuldade que é lidar com a não-convexidade. Deste modo, caso o item já seja convexo, sua representação resume-se a um conjunto unitário. Cada polígono convexo, por sua vez, é composto por um conjunto de vértices ordenados no sentido anti-horário, sendo que o primeiro vértice fornecido pelo usuário é tomado como o ponto de referência do item. É importante mencionar que a decomposição dos itens em polígonos convexos deve ser realizada pelo usuário em uma etapa de pré-processamento. Neste trabalho, fizemos uso de uma implementação, em linguagem C++, do algoritmo de programação dinâmica de Greene disponível no endereço eletrônico http://doc.cgal.org/latest/Partition_2/index.html. A Figura 31 ilustra o arquivo de dados de um item.

Figura 31 – Arquivo de dados de um item representado por dois polígonos convexos, a saber, P_0 e P_1 .

Fonte: Elaborada pelo autor.

Recipientes

Com relação aos recipientes, que são irregulares, representamos cada um deles através de seu menor retângulo envolvente com um conjunto de defeitos ([MARTINS; TSUZUKI, 2010](#)), como discutido na Seção 3.1. Visando simplificar a implementação, optamos por tratar os defeitos como se fossem zonas de pior qualidade, portanto não fazendo diferenciação entre os termos. Representamos cada zona de qualidade por um conjunto de polígonos convexos, assim como fizemos para os itens, e a cada uma atribuímos um valor inteiro não-negativo que indica seu nível de qualidade. Tal valor pode variar em uma escala que vai de 0 a $Q_{max} - 1$, em que 0 representa a pior qualidade e Q_{max} a qualidade máxima. O valor máximo que uma zona de qualidade pode receber é igual a $Q_{max} - 1$ (o qual representa a segunda melhor qualidade) devido ao fato de que, em nossa implementação, consideramos que a qualidade máxima é aquela encontrada na região complementar às zonas de qualidades, sendo a mesma para todos os recipientes. É

interessante deixar claro que, como estamos lidando com múltiplos recipientes, as zonas de um não necessariamente possuem os mesmos níveis de qualidade que as de outro; portanto, a escala deve ser definida considerando-se simultaneamente todos os níveis de qualidades de todos os recipientes. A Figura 32 traz um exemplo no qual podemos identificar quatro diferentes qualidades em um par de recipientes. Por fim, na Figura 33, temos um exemplo de arquivo de dados de um recipiente com uma única zona com qualidade 0.

Figura 32 – Identificando níveis de qualidades em um par de recipientes.

Fonte: Elaborada pelo autor.

Figura 33 – Arquivo de dados de um recipiente.

```
bin1.txt x
Bin 1
Width# 240.00
Height# 79.00
Number of quality zones# 1
-----
Zone# 1
Quality level# 0
Placement point# 204.19 24.43
Number of polygons# 2
number of vertices (polygon 0)# 6
vertices (x,y)#
0.00 0.00
-7.20 0.79
-9.60 2.37
-9.60 3.16
-4.80 3.16
-0.00 1.58
number of vertices (polygon 1)# 4
vertices (x,y)#
0.00 0.00
-2.40 -0.79
-9.60 -0.79
-7.20 0.79
```

Fonte: Elaborada pelo autor.

Dados gerais

Neste arquivo, inserimos informações mais gerais do problema, tais como: número de recipientes, identificação de cada recipiente, número de tipos de itens, e, para cada tipo de item, sua identificação, demanda, rotações permitidas e qualidade mínima exigida. A Figura 34 traz um exemplo de arquivo de dados gerais.

Figura 34 – Arquivo de dados gerais.

```
general_data.txt x
General Data

Number of bins# 3
ID# 1
ID# 2
ID# 3

Number of types of items# 4
ID# 1 demand# 9 number of rotations# 2 rotations# 0 180 minimum quality# 1
ID# 2 demand# 7 number of rotations# 2 rotations# 0 180 minimum quality# 2
ID# 3 demand# 9 number of rotations# 2 rotations# 0 180 minimum quality# 1
ID# 4 demand# 2 number of rotationa# 2 rotations# 0 180 minimum quality# 2

End General Data
```

Fonte: Elaborada pelo autor.

Nofit polygons

Para o cálculo dos *nofit polygons*, utilizamos o algoritmo de [Cuninghame-Green \(1989\)](#), sobre o qual discorreremos de forma detalhada na Seção 3.2.4. Como sabemos, uma vez que não pode haver sobreposição dos itens entre si, em uma fase de pré-processamento, calculamos os *nofit polygons* entre todos os pares de itens, incluindo cada item com ele mesmo (devido à sua demanda), em todas as rotações possíveis. Além disso, para evitar a sobreposição de itens com zonas de qualidades indesejáveis, também calculamos o *nofit polygon* entre cada item (orbital), em cada rotação possível, e cada zona, de cada recipiente, com qualidade inferior à mínima exigida para o item. Para cada *nofit polygon* calculado, armazenamos seus dados em um diferente arquivo, como mostram os exemplos da Figura 35. Sendo assim, o algoritmo, previamente, lê e armazena o *nofit polygon* de cada arquivo, e, durante a resolução do problema, recorre a eles conforme necessário.

Figura 35 – (a) *Nofit polygon* entre o item 7 na rotação 180°, e o item 2 na rotação 0° e (b) *Nofit polygon* entre a zona de qualidade 5 do recipiente 3, e o item 6 na rotação 0°.

Fonte: Elaborada pelo autor.

4.3.2 Dados de saída

Como saída, para cada recipiente, temos um arquivo contendo sua solução. A solução de um recipiente, por sua vez, é dada por uma lista que possui, em cada posição, as informações de cada item inserido (identificação, rotação, e ponto de alocação). A Figura 36 traz um exemplo de um arquivo contendo uma solução de um recipiente.

Figura 36 – Solução de um recipiente.

Fonte: Elaborada pelo autor.

EXPERIMENTOS COMPUTACIONAIS

Neste capítulo, apresentamos os experimentos computacionais realizados para os métodos desenvolvidos neste trabalho, os quais foram implementados nas linguagens de programação C/C++. Todos os experimentos foram feitos utilizando um computador com processador Intel Core i7-5500U de 2.40 GHz, 16 GB de memória RAM e sistema operacional Ubuntu 16.04 LTS.

Na Seção 5.1, descrevemos as instâncias utilizadas nos experimentos. Na Seção 5.2, apresentamos os resultados numéricos.

5.1 Instâncias

Devido ao fato do problema de pesquisa ter sido pouco abordado na literatura, não encontramos instâncias disponíveis para que pudéssemos realizar os testes. Sendo assim, criamos as nossas próprias fazendo adaptações em 15 instâncias clássicas do problema de corte de itens irregulares em faixa, cujas melhores soluções foram obtidas por [Elkeran \(2013\)](#). Antes de entrarmos em mais detalhes sobre como formamos o conjunto de instâncias, é importante adiantar que os recipientes utilizados nos testes foram gerados por um algoritmo desenvolvido por nós, chamado de *gerador de recipientes*, sobre o qual apresentamos mais detalhes a seguir. O algoritmo fornece ao usuário duas opções, ambas descritas a seguir.

1. Gerar recipientes irregulares, que, em nosso caso, são dados por retângulos com conjuntos de defeitos¹ posicionados em pontos estratégicos (Seção 3.1). Mais detalhadamente, a ideia do algoritmo é a seguinte: dados o comprimento e altura do retângulo, o mesmo insere de dois a três triângulos em cada um dos quatro cantos do retângulo, sem que haja sobreposição entre eles, começando a partir do canto inferior esquerdo e seguindo o sentido anti-horário, como ilustra o exemplo da Figura 37. Observe, ainda na Figura 37,

¹ Lembrando que tratamos defeitos como zonas de pior qualidade e que representamos cada zona de qualidade por um conjunto de polígonos convexos (ver Seção 4.3.1).

que os vértices dos triângulos são pontos pertencentes às bordas dos recipientes ou às hipotenusas dos triângulos retângulos².

Figura 37 – Recipiente obtido a partir do gerador de recipientes.

Fonte: Elaborada pelo autor.

2. Dado um recipiente, inserir zonas de qualidades. Neste caso, a partir de um conjunto de zonas de qualidades criado previamente, o algoritmo, aleatoriamente, seleciona quantas delas o usuário desejar, atribui a cada uma um nível de qualidade e, por fim, as fixa em pontos também aleatórios do recipiente (Figura 38). É importante enfatizar que, para criar o conjunto de zonas de qualidades mencionado, simplesmente selecionamos alguns itens das próprias instâncias utilizadas e os consideramos como tais. Com relação aos níveis de qualidades, uma vez que estamos lidando com múltiplos recipientes, o usuário precisa fornecer como dado de entrada um valor que expresse a qualidade máxima, Q_{max} , que os recipientes podem ter³. Por exemplo, caso este valor seja 4, isso significa que o algoritmo atribui a cada zona de cada recipiente algum valor entre 0 e 3, incluindo ambos, em que 0 representa a pior qualidade e 3 a segunda a melhor qualidade encontrada nos recipientes. É importante mencionar que cada um dos valores da escala é atribuído pelo menos uma vez a alguma das zonas de qualidades.

Figura 38 – Inclusão de zonas de qualidades em um recipiente.

Fonte: Elaborada pelo autor.

² Note que cada canto sempre terá um triângulo retângulo, sendo este necessariamente inserido primeiro.

³ Lembrando que a qualidade máxima é encontrada na região complementar às zonas de qualidades dos recipientes (ver Seção 4.3.1).

Voltando à discussão sobre como definimos as instâncias, com relação aos recipientes, utilizando o gerador, criamos quatro deles, sendo que todos possuem zonas com nível de qualidade igual a 0, 1 ou 2 (o que implica em $Q_{max} = 3$), representados pelas cores preta, cinza escuro e cinza claro, respectivamente, como mostra a Figura 39. Feito isso, para cada uma das 15 instâncias do problema de corte de itens irregulares em faixa escolhidas para adaptarmos, tomamos a altura do retângulo e o comprimento ótimo (ELKERAN, 2013) arredondado para o próximo valor inteiro, e redimensionamos os quatro recipientes gerados⁴. Deste modo, todas as instâncias passam a ter os mesmos recipientes, porém em tamanhos diferentes. Quanto aos itens, mantivemos seus dados originais, exceto pelas demandas⁵, e, aleatoriamente, atribuímos a cada um deles uma qualidade mínima, a qual varia em uma escala de 1 a 3 uma vez que os níveis de qualidades dos recipientes variam de 0 (pior qualidade) a 3 (melhor qualidade). Sendo assim, qualquer que seja o item, garantimos que o mesmo jamais irá sobrepor zonas cuja qualidade é considerada indesejável (inferior) para ele. Por exemplo, observando a Figura 39, caso seja exigido que um item possua uma qualidade mínima igual 2, então o mesmo não poderá ser posicionado sobre zonas com qualidades inferiores, representadas, de acordo com a escala, pelas cores preta e cinza escuro.

Figura 39 – Conjunto de recipientes gerados para cada uma das 15 instâncias.

Fonte: Elaborada pelo autor.

Devido ao fato de que para cada uma das 15 instâncias consideramos que o retângulo possui como dimensões a altura do retângulo original e comprimento ótimo obtido por (ELKERAN, 2013), sendo este último arredondado para o próximo inteiro, e, que, além disso, ainda temos as zonas de qualidades, obviamente que, em todos os testes realizados utilizando as demandas originais, qualquer que seja o método, um único recipiente não foi suficiente, sendo necessários no mínimo dois para empacotar todos os itens. Mais do que isso, constatamos que, na maioria das vezes, exatamente dois recipientes foram utilizados. Observe que tal fato nos impede de analisar a efetividade dos métodos, inclusive de realizar comparações entre os mesmos, uma vez que o objetivo é minimizar a quantidade de recipientes necessários para empacotar todos os itens; portanto, como alternativa, aumentamos as demandas originais, obtendo, assim, um maior

⁴ É importante enfatizar que redefinimos não apenas as dimensões dos retângulos, mas também as dimensões das zonas de qualidades, mantendo-as proporcionais ao tamanho dos retângulos.

⁵ Mais adiante, explicamos com mais detalhes o motivo de tal alteração.

número de itens. Para isso, dada uma instância qualquer, a ideia foi iterativamente aumentar em uma unidade a demanda de cada item, seguindo a ordenação por área decrescente, e, após cada acréscimo realizado, resolver a instância, utilizando cada um dos métodos com exceção daqueles que utilizam a estratégia *Random*, até surgir pelo menos um caso em que seja necessário utilizar mais um recipiente. Encontrado algum método para o qual tal situação ocorra, finalizamos o procedimento, obtendo, portanto, uma nova lista de demandas. No Apêndice A, apresentamos cada uma das instâncias, lembrando que os recipientes já foram bem definidos (Figura 39).

5.2 Resultados Numéricos

Nesta seção, apresentamos os resultados obtidos a partir da resolução das 15 instâncias utilizando cada um dos métodos desenvolvidos. Antes de entrar em detalhes, gostaríamos de mencionar que, durante a realização de testes iniciais, observamos de forma clara a inferioridade dos métodos que utilizam a estratégia de ordenação *QiDecArea_i* em relação aos demais. Sendo assim, daqui adiante, omitimos quaisquer informações relacionadas aos mesmos.

Os resultados, disponíveis a seguir, estão separados de acordo com os grupos de heurísticas: grupo *S* (Tabela 2); grupo *MD* (Tabela 3); grupo *MA* (Tabela 4); grupo *BL* (Tabela 5); e grupo *R* (Tabela 6). Para cada instância, reportamos o número de recipientes necessários para empacotar todos os itens ($\bar{m}$); a taxa de ocupação de cada recipiente utilizado ($ocp_{ID}(\%)$, $ID = 1, \dots, 4$); e, por fim, o tempo de resolução, em segundos ($t(s)$). É interessante enfatizar que a instância *Swim-z*, quando comparada às demais, apresentou um tempo computacional consideravelmente maior em todos os testes realizados. Isto se deu pelo fato de que, para esta instância em particular, foi necessário um maior número de polígonos convexos para representar cada item.

Começando pelo grupo *S*, note, na Tabela 2, que o método S_3 utilizou um recipiente a menos que o método S_1 para sete das 15 instâncias testadas, enquanto que este último reduziu o número de recipientes, também em uma unidade, apenas em uma instância quando comparado ao S_3 . Sendo assim, considerando este conjunto de instâncias, podemos concluir que o método S_3 é capaz de produzir melhores soluções que o S_1 .

Observando agora a Tabela 3, note que os métodos do grupo *MD*, isto é, MD_1 e MD_3 , quando comparados entre si, foram ambos capazes de reduzir o número de recipientes em uma unidade para duas instâncias. Ou seja, MD_1 reduziu o número de recipientes para as instâncias *Dighe1-z* e *Mao-z*, enquanto que MD_3 para as instâncias *Dagli-z* e *Dighe2-z*. Sendo assim, no que diz respeito à resolução deste conjunto de instâncias, podemos inferir apenas um empate entre ambos os métodos.

Tabela 2 – Resultados obtidos pelos métodos S_1 e S_3 para cada uma das 15 instâncias testadas.

Instância	S_1 (Spiral, User _r , DecArea _i)						S_3 (Spiral, DecArea _r , DecArea _i)					
	$\bar{m}$	ocp ₁ (%)	ocp ₂ (%)	ocp ₃ (%)	ocp ₄ (%)	t(s)	$\bar{m}$	ocp ₃ (%)	ocp ₁ (%)	ocp ₂ (%)	ocp ₄ (%)	t(s)
Albano-z	3	60,80	45,18	14,38	-	0,11	2	71,49	47,61	-	-	0,10
Dagli-z	2	71,60	57,42	-	-	0,09	2	67,81	59,78	-	-	0,08
Dighe1-z	4	59,02	47,63	13,46	20,13	0,02	3	50,55	48,19	40,06	-	0,02
Dighe2-z	4	59,17	51,39	27,41	12,48	0,01	3	59,38	50,45	39,54	-	0,01
Fu-z	3	63,12	41,38	4,45	-	0,01	2	68,46	39,20	-	-	0,02
Jakobs1-z	2	72,26	53,37	-	-	0,14	2	74,58	49,52	-	-	0,12
Jakobs2-z	3	66,82	52,50	3,48	-	0,21	2	73,82	47,52	-	-	0,22
Mao-z	2	65,89	45,13	-	-	0,36	2	57,86	51,96	-	-	0,32
Marques-z	3	72,94	47,63	7,12	-	0,19	2	64,33	62,16	-	-	0,16
Shapes0-z	3	55,33	39,16	1,00	-	0,15	3	56,38	36,54	1,44	-	0,13
Shapes1-z	3	56,70	45,28	4,48	-	0,22	3	59,52	44,63	1,14	-	0,21
Shapes2-z	2	65,28	47,10	-	-	0,07	2	64,60	46,45	-	-	0,07
Shirts-z	2	80,68	65,93	-	-	0,55	2	81,60	63,31	-	-	0,46
Swim-z	2	64,94	52,03	-	-	31,41	3	64,42	49,65	1,56	-	25,58
Trousers-z	3	71,51	42,17	4,42	-	0,23	2	76,22	40,44	-	-	0,24

Tabela 3 – Resultados obtidos pelos métodos MD_1 e MD_3 para cada uma das 15 instâncias testadas.

Instância	MD_1 (MinDistance, User _r , DecArea _i)						MD_3 (MinDistance, DecArea _r , DecArea _i)					
	$\bar{m}$	ocp ₁ (%)	ocp ₂ (%)	ocp ₃ (%)	ocp ₄ (%)	t(s)	$\bar{m}$	ocp ₃ (%)	ocp ₁ (%)	ocp ₂ (%)	ocp ₄ (%)	t(s)
Albano-z	3	60,80	45,18	14,31	-	0,15	3	66,47	45,15	7,60	-	0,18
Dagli-z	3	66,32	55,42	7,15	-	0,10	2	72,17	55,34	-	-	0,09
Dighe1-z	3	61,41	51,84	26,04	-	0,02	4	51,34	53,66	26,68	7,32	0,02
Dighe2-z	4	64,69	51,39	21,98	12,48	0,01	3	64,65	60,82	23,73	-	0,01
Fu-z	3	59,00	45,12	4,84	-	0,02	3	59,96	42,92	4,95	-	0,02
Jakobs1-z	3	73,20	48,11	4,23	-	0,16	3	76,56	42,12	5,41	-	0,13
Jakobs2-z	3	68,28	48,41	6,04	-	0,22	3	67,02	52,90	1,54	-	0,20
Mao-z	2	65,95	45,07	-	-	0,46	3	64,66	36,59	8,49	-	0,44
Marques-z	3	65,46	55,15	7,12	-	0,19	3	64,33	54,92	7,28	-	0,17
Shapes0-z	2	55,33	40,18	-	-	0,14	2	57,78	36,54	-	-	0,14
Shapes1-z	2	60,12	46,42	-	-	0,27	2	59,52	45,77	-	-	0,23
Shapes2-z	2	60,27	52,13	-	-	0,08	2	63,85	47,21	-	-	0,08
Shirts-z	2	80,19	66,42	-	-	0,54	2	82,50	62,38	-	-	0,52
Swim-z	2	64,94	52,03	-	-	34,69	2	66,72	48,86	-	-	36,08
Trousers-z	2	71,51	46,68	-	-	0,25	2	80,63	35,95	-	-	0,26

A Tabela 4, por sua vez, reporta os resultados obtidos pelos métodos MA_1 e MA_3 para cada uma das 15 instâncias. A partir da análise de tais resultados, podemos concluir a superperiodidade do método MA_1 em relação à MA_3 , uma vez que o primeiro foi capaz de utilizar um recipiente a menos que o segundo para quatro instâncias, enquanto que o segundo método melhorou a solução de apenas duas instâncias.

No que diz respeito aos métodos do grupo BL , cujos resultados estão disponíveis na Tabela 5, podemos claramente observar que o método BL_3 melhorou a solução de seis instâncias quando comparado ao BL_1 , o qual, por sua vez, melhorou a solução de apenas uma instância em relação à BL_3 . A partir deste fato, podemos afirmar que o método BL_3 é capaz de gerar soluções melhores que BL_1 para este conjunto de instâncias.

Tabela 4 – Resultados obtidos pelos métodos MA_1 e MA_3 para cada uma das 15 instâncias testadas.

Instância	MA_1 (MinArea, User _r , DecArea _i)						MA_3 (MinArea, DecArea _r , DecArea _i)					
	$\bar{m}$	ocp ₁ (%)	ocp ₂ (%)	ocp ₃ (%)	ocp ₄ (%)	t(s)	$\bar{m}$	ocp ₃ (%)	ocp ₁ (%)	ocp ₂ (%)	ocp ₄ (%)	t(s)
Albano-z	3	60,80	45,18	14,31	-	0,13	3	10,17	66,47	45,15	7,60	0,12
Dagli-z	2	71,60	57,42	-	-	0,10	2	69,24	58,32	-	-	0,09
Dighe1-z	3	61,41	51,84	26,04	-	0,02	4	51,34	53,66	26,68	7,32	0,02
Dighe2-z	4	64,69	51,39	21,98	12,48	0,02	3	64,65	60,82	23,73	-	0,02
Fu-z	2	63,73	45,32	-	-	0,02	2	68,46	39,20	-	-	0,01
Jakobs1-z	2	74,14	51,48	-	-	0,12	3	80,53	38,08	5,41	-	0,15
Jakobs2-z	3	68,13	53,20	1,51	-	0,24	3	66,87	53,06	1,54	-	0,23
Mao-z	2	65,89	45,13	-	-	0,43	3	64,66	36,59	8,49	-	0,44
Marques-z	3	65,46	55,15	7,12	-	0,18	2	77,67	48,59	-	-	0,19
Shapes0-z	2	55,33	40,18	-	-	0,16	2	55,37	38,99	-	-	0,15
Shapes1-z	2	60,12	46,42	-	-	0,28	3	57,96	43,95	3,43	-	0,27
Shapes2-z	2	60,27	52,13	-	-	0,08	2	63,85	47,21	-	-	0,07
Shirts-z	2	79,27	67,35	-	-	0,54	2	81,36	63,55	-	-	0,59
Swim-z	2	64,94	52,03	-	-	30,59	2	66,72	48,86	-	-	32,59
Trousers-z	2	71,51	46,68	-	-	0,28	2	80,63	35,95	-	-	0,33

Tabela 5 – Resultados obtidos pelos métodos BL_1 e BL_3 para cada uma das 15 instâncias testadas.

Instância	BL_1 (Bottom-Left, User _r , DecArea _i)						BL_3 (Bottom-Left, DecArea _r , DecArea _i)					
	$\bar{m}$	ocp ₁ (%)	ocp ₂ (%)	ocp ₃ (%)	ocp ₄ (%)	t(s)	$\bar{m}$	ocp ₃ (%)	ocp ₁ (%)	ocp ₂ (%)	ocp ₄ (%)	t(s)
Albano-z	3	64,15	54,46	1,93	-	0,13	3	66,91	45,27	7,02	-	0,14
Dagli-z	3	67,41	54,33	7,15	-	0,13	2	73,17	54,32	-	-	0,08
Dighe1-z	4	54,20	46,71	19,10	20,13	0,06	3	53,95	34,11	41,74	-	0,02
Dighe2-z	4	59,17	51,39	27,41	12,48	0,02	3	58,15	50,22	41,02	-	0,01
Fu-z	3	59,20	38,56	11,06	-	0,03	2	61,25	46,54	-	-	0,01
Jakobs1-z	2	67,41	58,24	-	-	0,15	2	73,52	50,59	-	-	0,11
Jakobs2-z	2	72,97	49,88	-	-	0,32	2	74,13	47,21	-	-	0,18
Mao-z	2	66,18	44,84	-	-	0,48	2	60,24	49,54	-	-	0,40
Marques-z	2	72,44	55,41	-	-	0,21	2	75,20	51,10	-	-	0,16
Shapes0-z	2	57,37	38,13	-	-	0,18	2	56,78	37,56	-	-	0,12
Shapes1-z	2	63,99	42,53	-	-	0,23	2	61,76	43,49	-	-	0,21
Shapes2-z	3	64,82	44,21	3,28	-	0,08	2	67,58	43,42	-	-	0,06
Shirts-z	3	78,06	63,62	4,84	-	0,56	2	79,33	65,61	-	-	0,41
Swim-z	2	68,41	48,55	-	-	42,79	3	63,46	48,35	3,85	-	36,01
Trousers-z	2	76,01	42,17	-	-	0,34	2	76,22	40,44	-	-	0,20

Resta-nos, portanto, analisar o desempenho dos métodos R_1 e R_3 . No caso deste grupo de métodos, uma vez que estamos lidando com uma estratégia que gera posicionamentos aleatórios para os itens nos recipientes, toda vez que resolvemos qualquer instância, uma diferente solução pode ser encontrada. Tendo isto em vista, implementamos os métodos deste grupo de modo que os mesmos, iterativamente, gerem 10 soluções e no final escolham a melhor entre elas, sendo considerada como melhor aquela que utiliza o menor número de recipientes possível. No caso de haver mais de uma solução com mesmo número mínimo de recipientes, o que é bastante provável, optamos por aquela cuja ocupação do último recipiente seja mínima, já que, quanto menor a ocupação, maior é a sobra, a qual pode possivelmente ser reutilizada. Caso ocorra empate também neste último critério, quando a quantidade mínima de recipientes é superior a dois, então olhamos para a taxa de ocupação do recipiente anterior. Por fim, para cada instância, executamos cada método 30 vezes e reportamos, na Tabela 6, o melhor resultado (b), o pior (w), e a média (av).

Tabela 6 – Resultados obtidos pelos métodos R_1 e R_3 para cada uma das 15 instâncias testadas.

Instância	R ₁ (Random, User _r , DecArea _i)							R ₃ (Random, DecArea _r , DecArea _i)					
	$\bar{m}$	$ocp_1(\%)$	$ocp_2(\%)$	$ocp_3(\%)$	$ocp_4(\%)$	$t(s)$	$\bar{m}$	$ocp_3(\%)$	$ocp_1(\%)$	$ocp_2(\%)$	$ocp_4(\%)$	$t(s)$	
Albano-z	<i>b</i>	2	67,64	52,93	-	-	0,85	2	74,78	44,26	-	-	1,03
	<i>w</i>	3	64,83	48,73	6,87	-	0,86	2	70,05	49,07	-	-	1,04
	<i>av</i>	2,87	64,67	51,68	4,14	-	0,85	2	72,97	46,10	-	-	1,03
Dagli-z	<i>b</i>	2	72,19	56,83	-	-	0,89	2	78,27	49,13	-	-	0,67
	<i>w</i>	2	69,11	59,92	-	-	0,91	2	72,92	54,58	-	-	0,65
	<i>av</i>	2	70,98	58,05	-	-	0,75	2	75,37	52,09	-	-	0,71
Dighe1-z	<i>b</i>	3	63,81	55,49	20,10	-	0,10	3	56,77	55,27	26,59	-	0,10
	<i>w</i>	3	63,81	55,42	20,17	-	0,11	3	58,45	51,25	28,92	-	0,12
	<i>av</i>	3	63,06	56,22	20,12	-	0,11	3	58,18	52,68	27,76	-	0,10
Dighe2-z	<i>b</i>	4	65,80	51,39	20,89	12,48	0,08	3	64,65	55,30	29,28	-	0,05
	<i>w</i>	4	47,32	51,39	39,06	12,48	0,09	3	64,65	47,72	36,89	-	0,06
	<i>av</i>	4	53,29	52,96	31,65	12,48	0,08	3	64,65	53,78	30,80	-	0,06
Fu-z	<i>b</i>	2	67,25	41,79	-	-	0,12	2	70,63	36,99	-	-	0,11
	<i>w</i>	2	64,13	44,92	-	-	0,16	2	65,30	42,42	-	-	0,13
	<i>av</i>	2	65,79	43,25	-	-	0,13	2	68,45	39,21	-	-	0,11
Jakobs1-z	<i>b</i>	2	77,91	47,70	-	-	1,41	2	80,53	43,46	-	-	1,05
	<i>w</i>	2	72,80	52,83	-	-	1,42	2	73,92	50,19	-	-	1,05
	<i>av</i>	2	75,10	50,52	-	-	1,42	2	77,57	46,48	-	-	1,28
Jakobs2-z	<i>b</i>	2	77,13	45,71	-	-	2,30	2	78,13	43,14	-	-	2,04
	<i>w</i>	2	73,05	49,80	-	-	2,30	2	73,67	47,68	-	-	2,05
	<i>av</i>	2	75,05	47,79	-	-	2,19	2	75,91	45,40	-	-	2,00
Mao-z	<i>b</i>	2	68,51	42,51	-	-	3,28	2	67,32	42,33	-	-	3,80
	<i>w</i>	2	65,95	45,07	-	-	4,28	2	60,14	49,64	-	-	3,88
	<i>av</i>	2	67,49	43,53	-	-	4,07	2	64,64	45,06	-	-	3,85
Marques-z	<i>b</i>	2	78,56	49,27	-	-	1,60	2	78,90	47,33	-	-	1,42
	<i>w</i>	2	73,53	54,32	-	-	1,99	2	74,33	51,99	-	-	1,40
	<i>av</i>	2	76,00	51,84	-	-	1,83	2	77,50	48,76	-	-	1,58
Shapes0-z	<i>b</i>	2	59,41	36,08	-	-	1,08	2	60,39	33,89	-	-	1,06
	<i>w</i>	2	55,73	39,77	-	-	1,09	2	56,38	37,97	-	-	1,07
	<i>av</i>	2	57,89	37,61	-	-	1,08	2	58,85	35,45	-	-	1,04
Shapes1-z	<i>b</i>	2	62,62	43,90	-	-	2,12	2	62,88	42,35	-	-	2,08
	<i>w</i>	3	60,80	44,59	1,12	-	2,23	2	58,41	46,91	-	-	2,08
	<i>av</i>	2,03	60,85	45,64	0,04	-	2,01	2	60,61	44,66	-	-	1,93
Shapes2-z	<i>b</i>	2	71,81	40,55	-	-	0,65	2	72,36	38,56	-	-	0,58
	<i>w</i>	2	64,67	47,71	-	-	0,75	2	68,18	42,81	-	-	0,72
	<i>av</i>	2	67,40	44,97	-	-	0,67	2	69,36	41,61	-	-	0,64
Shirts-z	<i>b</i>	2	84,00	62,60	-	-	3,30	2	83,24	61,63	-	-	3,35
	<i>w</i>	2	80,10	66,52	-	-	3,30	2	81,41	63,50	-	-	2,54
	<i>av</i>	2	81,69	64,92	-	-	3,14	2	82,22	62,68	-	-	2,88
Swim-z	<i>b</i>	2	69,43	47,52	-	-	355,50	2	69,62	45,91	-	-	262,24
	<i>w</i>	2	66,81	50,15	-	-	346,28	2	66,85	48,73	-	-	253,47
	<i>av</i>	2	68,09	48,86	-	-	318,96	2	68,15	47,41	-	-	263,89
Trousers-z	<i>b</i>	2	81,25	36,90	-	-	1,93	2	83,87	32,65	-	-	1,79
	<i>w</i>	2	71,51	46,68	-	-	1,95	2	80,63	35,95	-	-	1,79
	<i>av</i>	2	77,82	40,35	-	-	1,93	2	82,51	34,03	-	-	1,79

Ainda na Tabela 6, tomando como parâmetro as médias dos resultados obtidos a partir das 30 execuções dos métodos R_1 e R_3 para cada instância, podemos notar que, para 12 das 15 instâncias, ambos os métodos utilizaram a mesma quantidade de recipientes, porém, uma leve superioridade do método R_3 em relação à R_1 pode ser identificada uma vez que R_3 obteve melhores soluções para as demais instâncias (*Albano-z*, *Dighe2-z* e *Shapes1-z*). Outra informação interessante sobre o método R_3 , que pode ser observada em sua respectiva coluna $\bar{m}$, é o fato de que, para cada instância, a melhor e a pior solução utilizaram o mesmo número de recipientes, o que indica que o método não apresentou variações no número de recipientes utilizados para nenhuma das instâncias.

A seguir, na Tabela 7, reunimos os resultados (número de recipientes e tempo de resolução) dos métodos que obtiveram melhor desempenho. Fazendo uma análise geral, podemos facilmente notar a superioridade do método R_3 em relação aos demais, visto que o mesmo conseguiu utilizar apenas dois recipientes para aproximadamente 87% das instâncias e três para as demais, diferentemente dos outros métodos, que apresentaram maior variabilidade. É interessante enfatizar que a heurística R_3 obteve, inclusive, melhor desempenho que a heurística BL_3 , cuja estratégia de posicionamento de itens, *Bottom-Left*, constitui uma das mais utilizadas na literatura. No Apêndice B, apresentamos as ilustrações das melhores soluções obtidas pela heurística BL_3 para cada uma das 15 instâncias.

Tabela 7 – Resultados dos métodos que obtiveram melhor desempenho.

Instância	S_3		MD_1		MD_3		MA_1		BL_3		R_3 (média)	
	$\bar{m}$	$t(s)$	$\bar{m}$	$t(s)$	$\bar{m}$	$t(s)$	$\bar{m}$	$t(s)$	$\bar{m}$	$t(s)$	$\bar{m}$	$t(s)$
Albano-z	2	0,10	3	0,15	3	0,18	3	0,13	3	0,14	2	1,03
Dagli-z	2	0,08	3	0,10	2	0,09	2	0,10	2	0,08	2	0,71
Dighe1-z	3	0,02	3	0,02	4	0,02	3	0,02	3	0,02	3	0,10
Dighe2-z	3	0,01	4	0,01	3	0,01	4	0,02	3	0,01	3	0,06
Fu-z	2	0,02	3	0,02	3	0,02	2	0,02	2	0,01	2	0,11
Jakobs1-z	2	0,12	3	0,16	3	0,13	2	0,12	2	0,11	2	1,28
Jakobs2-z	2	0,22	3	0,22	3	0,20	3	0,24	2	0,18	2	2,00
Mao-z	2	0,32	2	0,46	3	0,44	2	0,43	2	0,40	2	3,85
Marques-z	2	0,16	3	0,19	3	0,17	3	0,18	2	0,16	2	1,58
Shapes0-z	3	0,13	2	0,14	2	0,14	2	0,16	2	0,12	2	1,04
Shapes1-z	3	0,21	2	0,27	2	0,23	2	0,28	2	0,21	2	1,93
Shapes2-z	2	0,07	2	0,08	2	0,08	2	0,08	2	0,06	2	0,64
Shirts-z	2	0,46	2	0,54	2	0,52	2	0,54	2	0,41	2	2,88
Swim-z	3	25,58	2	34,69	2	36,08	2	30,59	3	36,01	2	263,89
Trousers-z	2	0,24	2	0,25	2	0,26	2	0,28	2	0,12	2	1,79

Por fim, podemos concluir que todos os métodos resolveram as instâncias em um tempo computacional considerado baixo. Os métodos R_1 e R_3 , quando comparados aos demais, apresentaram um tempo computacional aproximadamente 10 vezes maior, o que era esperado, devido ao fato de executarem 10 iterações na busca por uma solução. Mas, como os tempos ainda assim são baixos, R_1 e R_3 continuam sendo considerados rápidos.

CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho, resolvemos a versão bidimensional do problema de corte de itens irregulares em recipientes, sendo que estes últimos também são irregulares e apresentam defeitos e zonas de qualidades. Como visto, este problema possui diversas aplicações em indústrias, sendo facilmente encontrado naquelas em que o couro é a principal matéria-prima. Além disso, possui grande importância não apenas do ponto de vista econômico, mas também do ponto de vista ambiental, uma vez que, quanto maior é o aproveitamento da matéria-prima, menor é a quantidade de sobras, as quais muitas vezes são simplesmente descartadas devido à sua impossibilidade de reutilização.

De modo a adquirirmos o embasamento necessário para a resolução do problema, fizemos uma extensa revisão da bibliografia, apresentada no Capítulo 3. A partir deste estudo, pudemos definir os modos como trataríamos algumas características importantes do problema. Ou seja, definimos que cada item seria representado por um conjunto de polígonos convexos; cada recipiente, por sua vez, seria representado por meio de seu menor retângulo envolvente com um conjunto de defeitos posicionados em pontos estratégicos; os defeitos e zonas de qualidades, assim como os itens, seriam representados também por conjuntos de polígonos convexos; por fim, concluímos que a não-sobreposição entre itens e a contenção dos mesmos no interior dos recipientes, ficariam garantidas pelos métodos do *nofit polygon* e *inner polygon*, respectivamente.

Para a resolução do problema, desenvolvemos um conjunto de heurísticas construtivas, encontradas no Capítulo 4, as quais diferem-se umas das outras em relação à ordenação dos itens, dos recipientes, e ao modo como buscamos por posicionamentos para os itens nos recipientes. Uma vez que não encontramos instâncias para o problema disponíveis na literatura, realizamos os experimentos computacionais, de modo a validar os métodos, utilizando um conjunto com 15 instâncias adaptadas do problema de corte de itens irregulares em faixa. Tais adaptações foram feitas de modo que pudéssemos ter múltiplos recipientes irregulares com defeitos e zonas de qualidades.

A partir dos resultados obtidos, disponíveis no Capítulo 5, concluímos que todos os métodos são capazes de resolver o problema em um tempo computacional considerado baixo. Além disso, pudemos facilmente notar a superioridade de um dos métodos em relação aos demais, uma vez que o mesmo foi capaz de produzir soluções que utilizam apenas dois recipientes, de um conjunto de quatro recipientes disponíveis, para aproximadamente 87% das instâncias e três recipientes para as demais, enquanto que os demais métodos apresentaram maior variabilidade.

Como trabalho futuro, sugerimos o desenvolvimento de uma metaheurística que gerencie as heurísticas aqui introduzidas visando a obtenção de melhores resultados. Além disso, também sugerimos a elaboração de modelos matemáticos para o problema e a resolução dos mesmos através de algum método exato utilizado na literatura.

REFERÊNCIAS

ALBANO, A.; SAPUPPO, G. Optimal allocation of two-dimensional irregular shapes using heuristic search methods. **Systems, Man and Cybernetics, IEEE Transactions on**, IEEE, v. 10, n. 5, p. 242–248, 1980. Citado na página 30.

ALVAREZ-VALDES, R.; MARTINEZ, A.; TAMARIT, J. M. A branch & bound algorithm for cutting and packing irregularly shaped pieces. **International Journal of Production Economics**, Elsevier, v. 145, n. 2, p. 463–477, 2013. Citado 2 vezes nas páginas 36 e 37.

ALVES, C.; BRÁS, P.; CARVALHO, J. M. V.; PINTO, T. A variable neighborhood search algorithm for the leather nesting problem. **Mathematical Problems in Engineering**, Hindawi Publishing Corporation, v. 2012, 2012. Citado 8 vezes nas páginas 18, 19, 22, 23, 24, 42, 43 e 53.

ALVES, C.; BRÁS, P.; CARVALHO, J. V. de; PINTO, T. New constructive algorithms for leather nesting in the automotive industry. **Computers & Operations Research**, Elsevier, v. 39, n. 7, p. 1487–1505, 2012. Citado 4 vezes nas páginas 18, 19, 42 e 43.

ART, R. C. **An approach to the two dimensional irregular cutting stock problem**. Tese (Doutorado) — Massachusetts Institute of Technology, USA, 1966. Citado 2 vezes nas páginas 30 e 38.

BABU, A. R.; BABU, N. R. A generic approach for nesting of 2-D parts in 2-D sheets using genetic and heuristic algorithms. **Computer-Aided Design**, Elsevier, v. 33, n. 12, p. 879–891, 2001. Citado 2 vezes nas páginas 21 e 22.

BALDACCI, R.; BOSCHETTI, M. A.; GANOVELLI, M.; MANIEZZO, V. Algorithms for nesting with defects. **Discrete Applied Mathematics**, Elsevier, v. 163, p. 17–33, 2014. Citado 9 vezes nas páginas 18, 19, 20, 21, 22, 24, 37, 38 e 42.

BENNELL, J. A.; OLIVEIRA, J. F. The geometry of nesting problems: A tutorial. **European Journal of Operational Research**, Elsevier, v. 184, n. 2, p. 397–415, 2008. Citado 5 vezes nas páginas 17, 27, 28, 29 e 32.

_____. A tutorial in irregular shape packing problems. **Journal of the Operational Research Society**, Nature Publishing Group, p. S93–S105, 2009. Citado 3 vezes nas páginas 32, 38 e 51.

CARLSSON, M.; OTTOSSON, G.; CARLSON, B. An open-ended finite domain constraint solver. In: SPRINGER. **Programming Languages: Implementations, Logics, and Programs**. [S.l.], 1997. p. 191–206. Citado na página 35.

CARRAVILLA, M. A.; RIBEIRO, C.; OLIVEIRA, J. F. Solving nesting problems with non-convex polygons by constraint logic programming. **International Transactions in Operational Research**, v. 10, n. 6, p. 651–663, 2003. Citado na página 35.

- CHERRI, L. H.; MUNDIM, L. R.; ANDRETTA, M.; TOLEDO, F. M.; OLIVEIRA, J. F.; CARRAVILLA, M. A. Robust mixed-integer linear programming models for the irregular strip packing problem. **European Journal of Operational Research**, Elsevier, v. 253, n. 3, p. 570–583, 2016. Citado 3 vezes nas páginas 22, 37 e 55.
- CRISPIN, A.; CLAY, P.; TAYLOR, G.; BAYES, T.; REEDMAN, D. Genetic algorithm coding methods for leather nesting. **Applied Intelligence**, Springer, v. 23, n. 1, p. 9–20, 2005. Citado 3 vezes nas páginas 18, 19 e 41.
- CUNINGHAME-GREEN, R. Geometry, shoemaking and the milk tray problem. **New Scientist**, v. 123, n. 1677, p. 50–53, 1989. Citado 3 vezes nas páginas 31, 32 e 57.
- DOWSLAND, K. A.; DOWSLAND, W. B.; BENNELL, J. A. Jostling for position: local improvement for irregular cutting patterns. **Journal of the Operational Research Society**, JSTOR, p. 647–658, 1998. Citado na página 39.
- ELKERAN, A. A new approach for sheet nesting problem using guided cuckoo search and pairwise clustering. **European Journal of Operational Research**, Elsevier, v. 231, n. 3, p. 757–769, 2013. Citado 5 vezes nas páginas 22, 23, 40, 59 e 61.
- FISCHETTI, M.; LUZZI, I. Mixed-integer programming models for nesting problems. **Journal of Heuristics**, Springer, v. 15, n. 3, p. 201–226, 2009. Citado 3 vezes nas páginas 34, 36 e 37.
- FWLER, R. J.; PATERSON, M. S.; TANIMOTO, S. L. Optimal packing and covering in the plane are NP-complete. **Information processing letters**, Elsevier, v. 12, n. 3, p. 133–137, 1981. Citado na página 35.
- GOMES, A. M.; OLIVEIRA, J. F. A 2-exchange heuristic for nesting problems. **European Journal of Operational Research**, Elsevier, v. 141, n. 2, p. 359–370, 2002. Citado 3 vezes nas páginas 39, 46 e 53.
- _____. Solving irregular strip packing problems by hybridising simulated annealing and linear programming. **European Journal of Operational Research**, Elsevier, v. 171, n. 3, p. 811–829, 2006. Citado na página 40.
- HEISTERMANN, J.; LENGAUER, T. The nesting problem in the leather manufacturing industry. **Annals of Operations Research**, Springer, v. 57, n. 1, p. 147–173, 1995. Citado 5 vezes nas páginas 18, 19, 22, 23 e 40.
- KONOPASEK, M. Mathematical treatments of some apparel marking and cutting problems. **US Department of Commerce Report**, v. 99, n. 26, p. 90857–10, 1981. Citado na página 28.
- LEAO, A. A.; TOLEDO, F. M.; OLIVEIRA, J. F.; CARRAVILLA, M. A. A semi-continuous mip model for the irregular strip packing problem. **International Journal of Production Research**, Taylor & Francis, v. 54, n. 3, p. 712–721, 2016. Citado na página 36.
- LEE, W.-C.; MA, H.; CHENG, B.-W. A heuristic for nesting problems of irregular shapes. **Computer-Aided Design**, Elsevier, v. 40, n. 5, p. 625–633, 2008. Citado 2 vezes nas páginas 18 e 41.
- MAHADEVAN, A. **Optimization in Computer-aided Pattern Packing (Marking, Envelopes)**. Tese (Doutorado) — North Carolina State University, USA, 1984. Citado na página 28.

- MARTINS, T. C.; TSUZUKI, M. S. G. Simulated annealing applied to the irregular rotational placement of shapes over containers with fixed dimensions. **Expert Systems with Applications**, Elsevier, v. 37, n. 3, p. 1955–1972, 2010. Citado 4 vezes nas páginas 23, 24, 53 e 55.
- OLIVEIRA, J. F.; GOMES, A. M.; FERREIRA, J. S. TOPOS – a new constructive algorithm for nesting problems. **OR-Spektrum**, Springer, v. 22, n. 2, p. 263–284, 2000. Citado 2 vezes nas páginas 39 e 53.
- OLIVEIRA, J. F. C.; FERREIRA, J. A. S. Algorithms for nesting problems. In: **Applied simulated annealing**. [S.l.]: Springer, 1993. p. 255–273. Citado 3 vezes nas páginas 22, 25 e 26.
- PINTO, T. M. P.; ALVES, C. M. M.; CARVALHO, J. M. V. V. D.; BRÁS, P. A. F. Heuristic methods for the leather nesting problem in the automotive industry. **International Journal of Business Excellence**, Inderscience Publishers (IEL), v. 9, n. 3, p. 332–347, 2016. Citado 3 vezes nas páginas 18, 19 e 43.
- PREPARATA, F. P.; SHAMOS, M. I. **Computational Geometry: An Introduction**. New York, NY: Springer-Verlag New York, 1985. Citado na página 28.
- RIBEIRO, C.; CARRAVILLA, M. A.; OLIVEIRA, J. F. Applying constraint logic programming to the resolution of nesting problems. **Pesquisa Operacional**, v. 19, n. 2, p. 239–247, 1999. Citado na página 35.
- SCHEITHAUER, G.; TERNO, J. Modeling of packing problems. **Optimization**, Taylor & Francis, v. 28, n. 1, p. 63–84, 1993. Citado na página 37.
- SEGENREICH, S. A.; BRAGA, L. M. P. F. Optimal nesting of general plane figures: A monte carlo heuristical approach. **Computers & graphics**, Elsevier, v. 10, n. 3, p. 229–237, 1986. Citado 4 vezes nas páginas 21, 22, 25 e 26.
- STOYAN, Y.; SCHEITHAUER, G.; GIL, N.; ROMANOVA, T. ϕ -functions for complex 2D-objects. **Quarterly Journal of the Belgian, French and Italian Operations Research Societies**, Springer, v. 2, p. 69–84, 2004. Citado na página 30.
- STOYAN, Y.; TERNO, J.; SCHEITHAUER, G.; GIL, N.; ROMANOVA, T. ϕ -functions for primary 2D-objects. **Studia Informatica Universalis**, v. 2, p. 1–32, 2001. Citado na página 29.
- TOLEDO, F. M. B.; CARRAVILLA, M. A.; RIBEIRO, C.; OLIVEIRA, J. F.; GOMES, A. M. The dotted-board model: a new mip model for nesting irregular shapes. **International Journal of Production Economics**, Elsevier, v. 145, n. 2, p. 478–487, 2013. Citado 3 vezes nas páginas 22, 36 e 37.
- YANG, X.-S.; DEB, S. Cuckoo search via lévy flights. In: IEEE. **Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on**. [S.l.], 2009. p. 210–214. Citado na página 40.
- ZHANG, Y.; YANG, C. A generic approach for leather nesting. In: IEEE. **Natural Computation, 2009. ICNC'09. Fifth International Conference on**. [S.l.], 2009. v. 5, p. 303–307. Citado 2 vezes nas páginas 18 e 41.

INSTÂNCIAS

A seguir, apresentamos as informações de cada uma das 15 instâncias utilizadas em nossos experimentos computacionais.

A.1 Albano-z

Nome: Albano-z

Retângulo: 9725 x 4900

Número total de itens: 25

Rotações permitidas: 0°

Tabela 8 – Demanda e qualidade mínima de cada tipo de item da instância Albano-z.

ID	1	2	3	4	5	6	7	8
Demanda	3	2	4	4	4	4	2	2
Qual. mín.	2	2	3	2	3	1	3	2

Figura 40 – Representação de cada tipo de item da instância Albano-z.

Fonte: Elaborada pelo autor.

A.2 Dagli-z

Nome: Dagli-z

Retângulo: 57 x 60

Número total de itens: 33

Rotações permitidas: 0° e 180°

Tabela 9 – Demanda e qualidade mínima de cada tipo de item da instância Dagli-z.

ID	1	2	3	4	5	6	7	8	9	10
Demanda	3	3	4	4	3	4	3	3	3	3
Qual. mín.	1	2	2	3	3	1	1	2	3	2

Figura 41 – Representação de cada tipo de item da instância Dagli-z.

Fonte: Elaborada pelo autor.

A.3 Dighe1-z

Nome: Dighe1-z

Retângulo: 100 x 100

Número total de itens: 17

Rotações permitidas: 0°

Tabela 10 – Demanda e qualidade mínima de cada tipo de item da instância Dighe1-z.

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Demanda	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1
Qual. mín.	1	3	1	1	3	3	3	3	1	1	2	2	3	3	2	3

Figura 42 – Representação de cada tipo de item da instância Dighe1-z.

Fonte: Elaborada pelo autor.

A.4 Dighe2-z

Nome: Dighe2-z

Retângulo: 100 x 100

Número total de itens: 11

Rotações permitidas: 0°

Tabela 11 – Demanda e qualidade mínima de cada tipo de item da instância Dighe2-z.

ID	1	2	3	4	5	6	7	8	9	10
Demanda	1	1	1	1	1	2	1	1	1	1
Qual. mín.	3	3	1	2	3	1	3	1	2	1

Figura 43 – Representação de cada tipo de item da instância Dighe2-z.

Fonte: Elaborada pelo autor.

A.5 Fu-z

Nome: Fu-z

Retângulo: 31 x 38

Número total de itens: 12

Rotações permitidas: 0°, 90°, 180° e 270°

Tabela 12 – Demanda e qualidade mínima de cada tipo de item da instância Fu-z.

ID	1	2	3	4	5	6	7	8	9	10	11	12
Demanda	1	1	1	1	1	1	1	1	1	1	1	1
Qual. mín.	3	3	2	3	1	3	2	3	3	2	1	1

Figura 44 – Representação de cada tipo de item da instância Fu-z.

Fonte: Elaborada pelo autor.

A.6 Jakobs1-z

Nome: Jakobs1-z

Retângulo: 11 x 40

Número total de itens: 27

Rotações permitidas: 0°, 90°, 180° e 270°

Tabela 13 – Demanda e qualidade mínima de cada tipo de item da instância Jakobs1-z.

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Demanda	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	2	1
Qual. mín.	3	2	1	2	3	3	3	1	3	3	1	1	3	2	3	2	2	3	2	1	3	3	2	2	2

Figura 45 – Representação de cada tipo de item da instância Jakobs1-z.

Fonte: Elaborada pelo autor.

A.7 Jakobs2-z

Nome: Jakobs2-z

Retângulo: 22 x 70

Número total de itens: 27

Rotações permitidas: 0°, 90°, 180° e 270°

Tabela 14 – Demanda e qualidade mínima de cada tipo de item da instância Jakobs2-z.

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Demanda	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	2
Qual. mín.	2	1	3	1	3	1	2	3	1	2	2	1	3	1	2	1	2	1	3	1	3	1	1	1	2

Figura 46 – Representação de cada tipo de item da instância Jakobs2-z.

Fonte: Elaborada pelo autor.

A.8 Mao-z

Nome: Mao-z

Retângulo: 1726 x 2550

Número total de itens: 21

Rotações permitidas: 0°, 90°, 180° e 270°

Tabela 15 – Demanda e qualidade mínima de cada tipo de item da instância Mao-z.

ID	1	2	3	4	5	6	7	8	9
Demanda	2	2	4	2	2	2	2	3	2
Qual. mín.	3	2	3	2	1	1	1	3	2

Figura 47 – Representação de cada tipo de item da instância Mao-z.

Fonte: Elaborada pelo autor.

A.9 Marques-z

Nome: Marques-z

Retângulo: 77 x 104

Número total de itens: 26

Rotações permitidas: 0°, 90°, 180° e 270°

Tabela 16 – Demanda e qualidade mínima de cada tipo de item da instância Marques-z.

ID	1	2	3	4	5	6	7	8
Demanda	5	4	2	2	4	4	3	2
Qual. mín.	1	3	3	1	2	2	2	2

Figura 48 – Representação de cada tipo de item da instância Marques-z.

Fonte: Elaborada pelo autor.

A.10 Shapes0-z

Nome: Shapes0-z
Retângulo: 58 x 40
Número total de itens: 59
Rotações permitidas: 0°

Tabela 17 – Demanda e qualidade mínima de cada tipo de item da instância Shapes0-z.

ID	1	2	3	4
Demanda	17	19	10	13
Qual. mín.	2	1	3	3

Figura 49 – Representação de cada tipo de item da instância Shapes0-z.

Fonte: Elaborada pelo autor.

A.11 Shapes1-z

Nome: Shapes1-z
Retângulo: 52 x 40
Número total de itens: 59
Rotações permitidas: 0° e 180°

Tabela 18 – Demanda e qualidade mínima de cada tipo de item da instância Shapes1-z.

ID	1	2	3	4
Demanda	17	19	10	13
Qual. mín.	2	1	2	3

Figura 50 – Representação de cada tipo de item da instância Shapes1-z.

Fonte: Elaborada pelo autor.

A.12 Shapes2-z

Nome: Shapes2-z

Retângulo: 26 x 15

Número total de itens: 31

Rotações permitidas: 0° e 180°

Tabela 19 – Demanda e qualidade mínima de cada tipo de item da instância Shapes2-z.

ID	1	2	3	4	5	6	7
Demanda	5	5	5	4	4	4	4
Qual. mín.	2	3	1	3	2	3	1

Figura 51 – Representação de cada tipo de item da instância Shapes2-z.

Fonte: Elaborada pelo autor.

A.13 Shirts-z

Nome: Shirts-z

Retângulo: 61 x 40

Número total de itens: 125

Rotações permitidas: 0° e 180°

Tabela 20 – Demanda e qualidade mínima de cada tipo de item da instância Shirts-z.

ID	1	2	3	4	5	6	7	8
Demanda	11	12	12	18	18	18	18	18
Qual. mín.	2	3	3	1	2	1	3	3

Figura 52 – Representação de cada tipo de item da instância Shirts-z.

Fonte: Elaborada pelo autor.

A.14 Swim-z

Nome: Swim-z

Retângulo: 5621 x 5752

Número total de itens: 59

Rotações permitidas: 0° e 180°

Tabela 21 – Demanda e qualidade mínima de cada tipo de item da instância Swim-z.

ID	1	2	3	4	5	6	7	8	9	10
Demanda	5	7	7	7	7	4	4	7	7	4
Qual. mín.	1	2	1	1	3	3	2	1	3	2

Figura 53 – Representação de cada tipo de item da instância Swim-z.

Fonte: Elaborada pelo autor.

A.15 Trousers-z

Nome: Trousers-z

Retângulo: 240 x 79

Número total de itens: 66

Rotações permitidas: 0° e 180°

Tabela 22 – Demanda e qualidade mínima de cada tipo de item da instância Trousers-z.

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Demanda	9	9	1	1	1	1	8	8	1	1	1	1	2	2	8	8	4
Qual. mín.	3	2	3	2	2	3	1	3	1	2	3	2	3	1	2	2	3

Figura 54 – Representação de cada tipo de item da instância Trousers-z.

Fonte: Elaborada pelo autor.

MELHORES SOLUÇÕES

A seguir, apresentamos as ilustrações das melhores soluções encontradas pela heurística R_3 para cada uma das 15 instâncias testadas em nosso trabalho. Lembrando que cada recipiente possui zonas com nível de qualidade igual a 0, 1 ou 2, representados pelas cores preta, cinza escuro e cinza claro, respectivamente, sendo uma qualidade melhor do que a outra nesta respectiva ordem. Quanto aos itens, de modo a distinguí-los com relação às qualidades mínimas, definimos três diferentes tons de azul, sendo o tom mais escuro utilizado para representar itens cuja qualidade mínima seja igual a 1, o tom intermediário para itens cuja qualidade mínima seja igual a 2 e, por fim, o tom mais claro para representar itens cuja qualidade mínima seja igual a 3.

Figura 55 – Albano-z.

Fonte: Elaborada pelo autor.

Figura 56 – Dagli-z.

Fonte: Elaborada pelo autor.

Figura 57 – Dighe1-z.

Fonte: Elaborada pelo autor.

Figura 58 – Dighe2-z.

Fonte: Elaborada pelo autor.

Figura 59 – Fu-z.

Fonte: Elaborada pelo autor.

Figura 60 – Jakobs1-z.

Fonte: Elaborada pelo autor.

Figura 61 – Jakobs2-z.

Fonte: Elaborada pelo autor.

Figura 62 – Mao-z.

Fonte: Elaborada pelo autor.

Figura 63 – Marques-z.

Fonte: Elaborada pelo autor.

Figura 64 – Shapes0-z.

Fonte: Elaborada pelo autor.

Figura 65 – shapes1-z.

Fonte: Elaborada pelo autor.

Figura 66 – Shapes2-z.

Fonte: Elaborada pelo autor.

Figura 67 – Shirts-z.

Fonte: Elaborada pelo autor.

Figura 68 – Swim-z.

Fonte: Elaborada pelo autor.

Figura 69 – Trousers-z.

Fonte: Elaborada pelo autor.