UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Uma integração dos problemas de empacotamento de peças irregulares e de caminho mínimo de corte

Larissa Tebaldi de Oliveira

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

ICMC USP
SÃO CARLOS

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Larissa Tebaldi de Oliveira

Uma integração dos problemas de empacotamento de peças irregulares e de caminho mínimo de corte

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

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Franklina Maria Bragion de Toledo
Coorientador: Prof. Dr. José Fernando da Costa Oliveira

USP – São Carlos
Maio de 2019

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

O48i

Oliveira, Larissa Tebaldi de
Uma integração dos problemas de empacotamento de peças irregulares e de caminho mínimo de corte / Larissa Tebaldi de Oliveira; orientadora Franklina Maria Bragion de Toledo; coorientador José Fernando da Costa Oliveira. -- São Carlos, 2019.
134 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2019.

1. Empacotamento de Peças Irregulares. 2. Caminho de Corte. 3. Modelos Integrados. 4. Matheurísticas. I. Toledo, Franklina Maria Bragion de, orient. II. Oliveira, José Fernando da Costa, coorient. III. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Larissa Tebaldi de Oliveira

Integrating nesting and cutting path determination problems

Doctoral dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Profa. Dra. Franklina Maria Bragion de Toledo
Co-advisor: Prof. Dr. José Fernando da Costa Oliveira

USP – São Carlos
May 2019

Aos meus pais e noivo.

AGRADECIMENTOS

Gostaria de agradecer à minha orientadora, Profª. Drª. Franklina M. B. Toledo, ao meu coorientador Prof. Dr. José Fernando Oliveira e à Profª. Drª Maria Antónia Carravilla pela oportunidade dada e por toda paciência, amizade, atenção, tempo, sugestões e esforços empenhados durante todo o período deste trabalho.

Aos meus pais Maria Carolina e Cacildo, que sempre me incentivaram a estudar e me apoiaram ao longo da vida.

Ao meu noivo Renato, pelo seu incondicional apoio, amizade, carinho e, principalmente, sua paciência.

A Everton Fernandes da Silva pela parceria e a Luiz Henrique Cherri e Marcos Rodrigues Okamura, que assim como Everton, me ajudaram ao longo desse trabalho.

Aos professores e funcionários do Instituto de Ciências Matemáticas e de Computação (ICMC/USP) e da Faculdade de Engenharia da Universidade do Porto (FEUP).

A todos os meus colegas do Laboratório de Otimização do ICMC/USP (LOt) e do laboratório do INESC/TEC, Portugal.

E à Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP - pelo apoio financeiro concedido sob os processos 13/25743-6 (bolsa país), 16/09476-6 (bolsa BEPE), 10/10133-0 (projeto temático) e 13/07375-0 (CEPID-CeMEAI).

“A ciência nunca resolve um problema sem criar pelo menos outros dez.”

(George Bernard Shaw)

RESUMO

OLIVEIRA, L. T. **Uma integração dos problemas de empacotamento de peças irregulares e de caminho mínimo de corte**. 2019. 134 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Presente em diversos processos industriais, que variam desde pequenas confecções até grandes indústrias da área de metal-mecânica, os problemas de empacotamento visam definir o posicionamento de itens menores sobre objetos maiores minimizando, frequentemente, a perda de material utilizado. O problema de empacotamento de peças irregulares em faixas, estudado nesta pesquisa, tem como principal característica, e obstáculo, possuir itens irregulares. Em algumas indústrias surge, após a determinação do empacotamento, um segundo problema: a determinação do caminho mínimo de corte. Embora a solução do primeiro influencie fortemente a resolução do segundo, não é de nosso conhecimento que existam, até o momento, estratégias que integrem esses problemas. Neste trabalho, são propostos dois modelos integrados de empacotamento de peças irregulares e caminho de corte. O primeiro modelo busca minimizar o caminho de corte entre as peças considerando um ponto fixo de início de corte (vértice fixo) para cada peça, enquanto que o segundo considera o corte por peça a partir de um vértice qualquer das peças. Testes computacionais mostram que é vantajosa a integração dos problemas contudo, como ambos são problemas de difícil solução, o problema integrado é pelo menos tão difícil quanto os problemas isolados, logo apenas instâncias de pequeno porte foram resolvidas de forma exata. Uma matheurística, baseada no algoritmo genético de chaves aleatórias viciadas, é proposta para o problema de empacotamento de peças irregulares em faixa em domínio contínuo e, em seguida, estendida para o problema integrado. Os resultados são promissores, pois a matheurística consegue encontrar solução para instâncias que não haviam sido resolvidas através dos modelos integrados previamente propostos.

Palavras-chave: Empacotamento de Peças Irregulares, Caminho de Corte, Modelos Integrados, Matheurísticas.

ABSTRACT

OLIVEIRA, L. T. **Integrating nesting and cutting path determination problems**. 2019. 134 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Having great applicability in industries, ranging from small clothing industries to large metal mechanic ones, packing problems aim to determine the positioning of small pieces over a large object minimizing, for instance, raw material waste. The main characteristic and obstacle of the irregular strip packing problem, studied in this research, is the irregular shape of its pieces. In some industries, after a layout of pieces has been defined, a second problem arises: the cutting path determination problem. Although the solution of the first strongly influences the resolution of the second, to the best of our knowledge, there are no strategy to integrate these problems. Here, we propose two irregular strip packing and cutting path integrated models. The first one minimizes the cutting path between the pieces considering that the cutting starts at a fixed vertex for each piece, while the second considers the cutting start point in any vertex of the pieces. Computational tests show that it is advantageous to integrate the problems, however, as both are difficult to solve, the integrated one is at least as difficult as each of them, so only small instances were solved to optimality. A matheuristic, based on the biased random-key genetic algorithm, is proposed for the continuos irregular strip packing problem and then extended to the integrated problem. The results are promising, the matheuristics is able to find solution for instances that had not been solved through the previously proposed integrated models.

Keywords: Irregular Packing Problem, Cutting Path, Integrated Models, Matheurísticas.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

Em diversas empresas, como fábricas de móveis, calçados e roupas, os produtos finais são fabricados a partir da junção de várias partes (peças) que foram previamente cortadas de objetos maiores (placas). Por exemplo, uma camisa é formada pelas peças que compõem o tronco, as mangas, os punhos e o colarinho. Um dos grandes desafios enfrentados por estas empresas é buscar formas de reduzir a perda de material que ocorre durante a etapa de corte. Frequentemente, encontramos na literatura trabalhos que estudam este problema, classificando-o como um problema de empacotamento de peças regulares/irregulares em placas regulares/irregulares, dependendo da geometria das peças e das placas.

O *Problema de Empacotamento de Peças Irregulares em Faixas*, estudado neste trabalho, também conhecido como *Irregular Strip Packing Problem*, consiste em agrupar peças de formato irregular em uma placa de altura fixa e comprimento ilimitado. O objetivo é minimizar o comprimento utilizado da placa. Devido à sua aplicabilidade e dificuldade de resolução (Fowler, Paterson e Tanimoto (1981)), este problema vem, há algumas décadas, ganhando o interesse de pesquisadores da área de Pesquisa Operacional. Na classificação proposta por Wäscher, Haußner e Schumann (2007), ele é definido como bidimensional e irregular, e sua principal característica, que o distingue dos demais problemas de empacotamento, é a geometria irregular de suas peças, que podem ser convexas ou não convexas.

Uma vez definido o empacotamento das peças, surge, em algumas indústrias, a questão adicional de encontrar o melhor modo de cortar as peças, ou seja, o *Problema de Determinar um Caminho de Corte* a ser seguido para executar o empacotamento anteriormente definido. Este problema consiste em definir a trajetória que uma ou mais ferramentas de corte devem seguir de forma a cortar um dado empacotamento. O objetivo pode ser, por exemplo, reduzir o tempo total de corte, e assim possibilitar a execução de um maior número de empacotamentos ao final de um turno de trabalho, ou reduzir o trajeto de corte, importante quando a ferramenta tem custo elevado e se desgasta ao longo do processo de corte, como é o caso do corte com diamante

artificial.

Manber e Israni (1984), Imahori *et al.* (2008), Sherif, Jawahar e Balamurali (2014) e Anand e Babu (2015) destacam que as decisões tomadas na etapa do empacotamento influenciam fortemente a resolução do problema de caminho mínimo de corte, no entanto, pouca atenção tem sido dada ao empacotar as peças em uma perspectiva de seu corte. Um empacotamento com comprimento de caminho de corte menor pode melhorar a produtividade de um dia de trabalho, pois o tempo economizado durante o processo de corte pode ser usado para cortar mais placas. Por outro lado, isso pode levar a um maior desperdício de material. É de nosso conhecimento que, até o momento, apenas Anand e Babu (2015) abordam esses problemas de maneira integrada, porém apenas com peças regulares (retangulares). Assim, o principal objetivo deste trabalho é mostrar se é vantajoso integrar os problemas de empacotamento de peças irregulares em faixas e de determinação de caminho mínimo de corte.

Existem, em situações reais, dois tipos de cortes: por peças e por arestas. No primeiro caso, por questões técnicas, as peças devem ser cortadas por inteiro, ou seja, uma peça deve ser completamente cortada para que o corte da próxima peça seja iniciado. Esta situação aparece frequentemente na indústria têxtil. No segundo, as peças podem ser cortadas por arestas, ou seja, as arestas das peças podem ser cortadas de forma independente uma das outras, não havendo a necessidade de cortar todas as arestas de uma peça antes que o corte de uma outra peça seja iniciado. Este é um processo usual nas indústrias metal-mecânica. Desta forma, duas versões para o problema integrado foram consideradas neste trabalho. Na primeira, o problema é considerado discreto, ou seja, o empacotamento das peças é realizado em um objeto definido por uma malha e o corte é realizado por peças, a partir de um de seu vértices. Enquanto que na segunda, os problemas são tratados de forma contínua, isto é, o posicionamento das peças no objeto é livre e as peças podem ser cortadas por suas arestas.

Para a primeira versão do problema integrado, propomos uma modelagem discreta para o empacotamento das peças considerando que o corte é feito por peça a partir de um vértice pré-determinado, ou seja, cada peça é representada por um ponto fixo pelo qual a ferramenta de corte deve iniciar e concluir o corte da peça. Em seguida, esse modelo é estendido para considerar um maior número de possibilidades para o início do corte, ou seja, o modelo permite que todos os vértices da peça sejam alternativas para o início de seu corte.

Como ambos os problemas de empacotamento e de caminho de corte são de difícil resolução, para a segunda versão do problema integrado propomos uma matheurística. Primeiramente, desenvolvemos uma matheurística baseada no BRKGA com busca local para o problema de empacotamento de peças irregulares em faixa. Por fim, com base nessa matheurística, propomos uma matheurística para o problema integrado, incorporando o modelo de determinação de caminho de corte por aresta proposto por Silva (2016).

1.1 Organização do trabalho

Esta tese possui nove capítulos, estruturados da seguinte maneira: no Capítulo 2 apresentamos uma revisão bibliográfica do problema de empacotamento de peças irregulares, seguida, no Capítulo 3, de uma revisão do problema de caminho de corte e, no Capítulo 4, dos trabalhos que buscam integrar os problemas de empacotamento e caminho de corte. Nos Capítulos 5 e 6, abordamos a primeira versão do problema integrado, ou seja, integramos os problemas de empacotamento e caminho de corte considerando que o empacotamento é feito em domínio discreto e o corte por peças sendo que, no Capítulo 5 propomos um modelo discreto, em que o corte da peça é realizado a partir de um ponto fixo, pelo qual a ferramenta de corte deve iniciar e concluir o corte da peça e no Capítulo 6, estendemos esse modelo para permitir que todos os vértices da peça sejam considerados como alternativa para o início do corte da peça. Nos dois capítulos seguintes, consideramos a segunda versão do problema integrado, em que o empacotamento é feito em domínio contínuo e o corte é realizado por arestas sendo que, no Capítulo 7 desenvolvemos uma matheurística baseada no BRKGA e busca local para o problema de empacotamento de peças irregulares em faixa e no Capítulo 8 estendemos essa matheurística para o problema integrado, incorporando o modelo de determinação de caminho de corte por aresta. Por fim, as conclusões e os trabalhos futuros são apresentados no Capítulo 9.

CAPÍTULO

2

O PROBLEMA DE EMPACOTAMENTO

Neste capítulo é apresentada uma revisão bibliográfica sobre o problema de empacotamento de peças irregulares. Uma definição do problema é dada na Seção 2.1, as estratégias utilizadas no tratamento geométrico do problema são resumidas na Seção 2.2 e os métodos de resolução estudados são descritos na Seção 2.3.

2.1 Caracterização do problema

O problema do empacotamento de peças irregulares em faixas consiste em alocar peças (objetos menores) de formatos irregulares em uma placa (objeto maior) regular de altura fixa e comprimento ilimitado, com o objetivo de minimizar o comprimento de placa utilizado. Na Figura 1 é ilustrado o posicionamento de peças irregulares em uma faixa de tecido, para a fabricação de roupas de banho.

Figura 1 – Empacotamento de peças irregulares.

Fonte: Bennell e Oliveira (2008).

Existem casos em que as peças podem ser livremente rotacionadas, possuir ângulos específicos de rotação (90º, 180º, ...) ou então não possuir rotação alguma. Contudo, em todos os casos, as seguintes restrições devem ser respeitadas:

- todas as peças alocadas na placa devem estar inteiramente contidas na mesma,
- não pode haver sobreposição entre peças, e
- no caso de existir demanda, ela deve ser atendida.

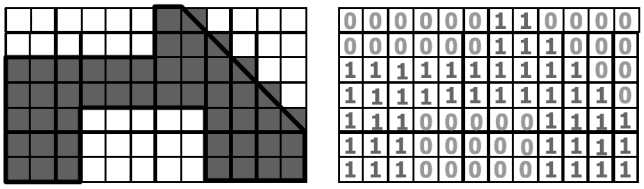
Embora o problema de empacotamento de peças irregulares em faixas seja simples de ser compreendido, o tratamento geométrico das peças e da placa, principal característica e obstáculo do problema, acrescenta uma grande dificuldade para sua resolução.

2.2 Aspectos geométricos

Uma das dificuldades computacionais associada à geometria das peças é determinar automaticamente quando duas peças se sobrepõem e garantir que elas estejam totalmente contidas na placa. Bennell e Oliveira (2008) discutem estratégias que podem ser utilizadas para representar a parte geométrica deste problema, que são: *raster points*, trigonometria direta, *no-fit polygon* (NFP) e função *phi*.

A estratégia *raster points*, proposta por Oliveira e Ferreira (1993), reduz as informações geométricas do problema através da discretização da placa e das peças utilizando uma malha, computacionalmente representada por uma matriz. Há, na literatura, três representações desenvolvidas para esta abordagem. A mais simples, chamada de *raster* 0-1, atribui valor 1 aos elementos da matriz correspondentes ao interior de uma peça e 0 caso contrário, conforme ilustrado na Figura 2.

Figura 2 – A representação *raster* 0-1 para peças irregulares.



Fonte: Bennell e Oliveira (2008).

Alocar uma peça utilizando esta estratégia se resume a adicionar 0 ou 1 aos elementos da matriz que representa a placa. Além disso, para verificar se há sobreposição entre peças basta verificar se algum elemento da matriz da placa possui valor maior que 1, o que significa que há mais de uma peça ocupando a mesma posição da placa.

Além da facilidade de implementação, esta estratégia possui a vantagem de poder representar tanto peças convexas como não convexas. Contudo, ela não é capaz de representar com exatidão peças com arestas não ortogonais e possui um alto consumo de memória durante o processamento das informações.

No caso de peças com arestas não ortogonais, uma alternativa é utilizar polígonos para representar as peças, ou seja, trigonometria direta. Basicamente, essa estratégia exige testes para verificar a interseção de arestas e identificar casos onde as peças se sobrepõem parcialmente, e testes para verificar a inclusão de pontos, a fim de verificar se uma peça está contida inteiramente em outra. Comparada ao *raster points*, a precisão na representação da peça utilizando trigonometria direta é melhor, porém o esforço para verificar a factibilidade no posicionamento de duas peças torna-se exponencial em relação ao total de arestas das peças.

A função *phi* é a mais recente ferramenta para o tratamento geométrico. Seu objetivo é representar por meio de expressões matemáticas as posições relativas de duas peças. Se o valor da função *phi* for maior que 0 significa que as peças estão separadas, se o valor é igual a 0 significa que elas se tocam, e se o valor é menor que 0 então existe sobreposição entre elas. Bennell e Oliveira (2008) descrevem as funções *phi* para peças de formatos primários, como círculos, retângulos e polígonos regulares, propostas por Stoyan *et al.* (2002) e Stoyan *et al.* (2004). Os autores citam que peças de formatos não primários podem ser representadas pela união ou interseção de formas primárias. Porém, a não existência de processos algorítmicos para a geração de funções *phi* para quaisquer polígonos ainda é uma questão em aberto nesta abordagem.

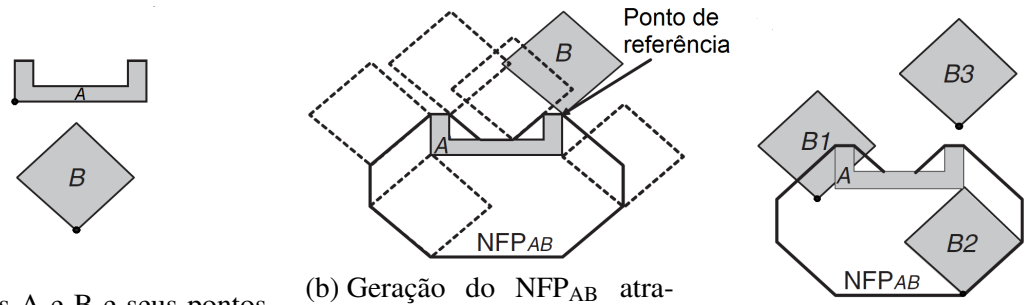
Por fim, o *no-fit polygon* (NFP) é um polígono obtido a partir de dois outros polígonos e é utilizado para determinar se estes se sobrepõem, se tocam ou se estão separados. Nesta estratégia, todas as peças são representadas por polígonos e possuem um ponto de referência. Na Figura 3a são ilustradas duas peças e seus respectivos pontos de referência. Vale ressaltar que os pontos de referência não precisam ser vértices do polígono.

Dentre as formas de se obter o NFP entre duas peças citamos: o algoritmo de deslizamento, a soma de Minkowski e a decomposição, para mais detalhes veja Bennell e Oliveira (2008). Na Figura 3b é apresentada a geração de um NFP através do algoritmo de deslizamento. Neste caso, consideramos que uma peça está fixa (Peça A) enquanto a outra (Peça B) orbita ao seu redor, sempre tocando-a, porém nunca se sobrepondo. O polígono traçado pelo ponto de referência da peça orbital dá origem ao NFP entre as peças A e B, denotado por $\mathrm{NFP}_{\mathrm{AB}}$.

Após obtido o NFP, utiliza-se um teste simples onde é identificado se o ponto de referência da peça a ser adicionada ao empacotamento (peça orbital) está dentro, fora ou na fronteira do NFP. Na Figura 3c é ilustrado um exemplo. Nesse caso, temos que a Peça A já está alocada, logo se o ponto de referência da Peça B não pertence ao $\mathrm{NFP}_{\mathrm{AB}}$ então garantimos que as peças A e B estão afastadas (Peça B3); porém se o ponto de referência da Peça B pertence a fronteira do $\mathrm{NFP}_{\mathrm{AB}}$, então temos que as peças se tocam (Peça B2); já se o ponto de referência da Peça B pertence ao interior do $\mathrm{NFP}_{\mathrm{AB}}$, então as peças se sobrepõem (Peça B1).

Alguns autores, como Cherri *et al.* (2016), utilizam as peças não convexas divididas em partes convexas. Na Figura 4a é ilustrado um exemplo, em que a Peça A (não convexa) é dividida em 3 partes convexas. Nesse caso, o $\mathrm{NFP}_{\mathrm{AB}}$ passa a ser composto pela interseção dos

Figura 3 – Ilustração do *no-fit polygon* entre duas peças.



(a) Peças A e B e seus pontos de referência.

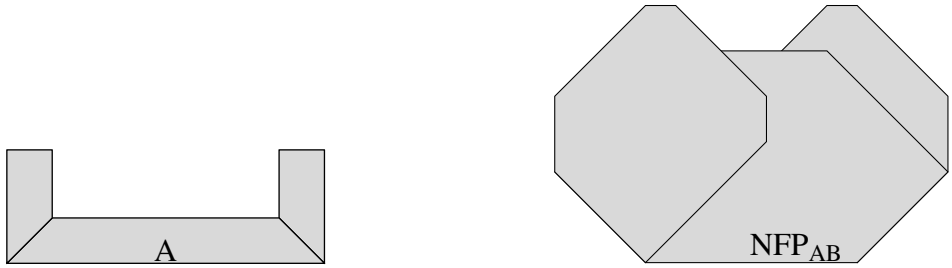
(b) Geração do NFP_{AB} através do algoritmo de deslizamento.

(c) Posicionamento da Peça B.

Fonte: Adaptada de Bennell e Oliveira (2008).

NFPs entre as partes da Peça A e a Peça B, como ilustrado na Figura 4b. O raciocínio é análogo para o caso em que ambas as peças A e B são peças não convexas divididas em partes convexas.

Figura 4 – Peça e NFP divididos em partes.



(a) Peça dividida em partes convexas.

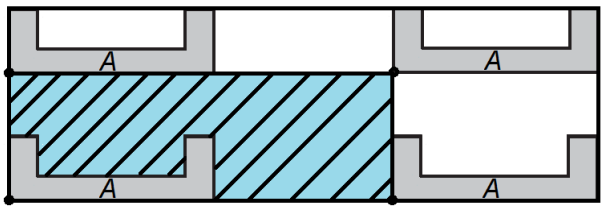
(b) *No-fit polygon* de uma peça divida em partes.

Fonte: Elaborada pela autora.

Uma das vantagens em se utilizar o NFP é a redução do processamento computacional, uma vez que eles são calculados durante a etapa de pré-processamento, ou seja, os NFPs são calculados uma única vez e fornecidos como dados de entrada. Assim, para analisar a relação entre duas peças basta verificar se o ponto de referência da peça a ser adicionada está dentro, fora ou na fronteira do NFP das peças. Porém, desenvolver um gerador de NFP robusto é uma tarefa custosa e não trivial.

Utilizando um conceito muito parecido ao NFP temos o *inner-fit polygon* (IFP). O IFP, também conhecido como *inner-fit rectangle* no caso em que as placas são retangulares, é a região da placa onde é possível alocar o ponto de referência de uma peça de forma que fique garantido que ela está inteiramente contida dentro da placa. Na Figura 5 é ilustrado o IFP da Peça A, IFP_A, representado pela região hachurada da figura. Este retângulo é desenhado pelo ponto de referência da Peça A ao deslocá-la dentro da placa, sempre garantindo que a peça nunca ultrapasse os limites da placa.

Figura 5 – *Inner-fit polygon* da Peça A.



Fonte: Elaborada pela autora.

2.3 Métodos de resolução

De acordo com Fowler, Paterson e Tanimoto (1981), o problema de empacotamento de peças irregulares é NP-completo. Isto justifica o atual cenário da área, no qual há um grande número de artigos que utilizam heurísticas e/ou meta-heurísticas para tratar o problema, enquanto poucos métodos exatos foram propostos. Em Bennell e Oliveira (2009), encontramos um tutorial sobre as abordagens de resolução propostas na literatura para o problema, que foi recentemente atualizado em Álvarez-Valdés, Carravilla e Oliveira (2018).

De forma geral, os métodos heurísticos propostos podem ser divididos em dois grupos:

1. **heurísticas construtivas**: têm por objetivo construir um empacotamento e obter uma solução factível para o problema, veja por exemplo Art (1966); Albano e Sapuppo (1980); Segenreich e Braga (1986); Dowsland, Vaid e Dowsland (2002); Burke *et al.* (2006) e Bennell e Song (2008); e

2. **heurísticas de melhoria**: têm por objetivo melhorar soluções já existentes, veja por exemplo Oliveira e Ferreira (1993); Jakobs (1996); Dowsland, Dowsland e Bennell (1998); Oliveira, Gomes e Ferreira (2000); Babu e Babu (2001); Bennell e Dowsland (2001); Gomes e Oliveira (2002); Takahara, Kusumoto e Miyamoto (2003); Burke *et al.* (2006) e Elkeran (2013).

Uma das heurística mais famosa da área é a *Bottom-Left* (BL). Utilizada tanto individualmente como em conjunto com outros métodos, a BL é um heurística construtiva em que os itens são alocados mais à esquerda e mais abaixo possível da placa de acordo com uma sequência preestabelecida. O método heurístico que obteve as melhores soluções até o momento foi proposto por Elkeran (2013). Neste método, as peças, cujos formatos são polígonos congruentes, são agrupadas aos pares e, em seguida, um método que combina a meta-heurística *cuckoo search* com uma busca local direcionada é aplicado.

Alguns dos trabalhos mais recentes da literatura utilizam o algoritmo genético de chaves aleatórias viciadas (BRKGA - do inglês *Biased Random-Key Genetic Algorithm*). No BRKGA, cada solução é representada por um vetor de n chaves aleatórias. A transformação deste vetor em

uma solução do problema é feita pelo decodificador, que também retorna um valor representativo da solução decodificada. Mundim, Queiroz e Andretta (2016) utilizam uma abordagem baseada no BRKGA para resolver dois problemas: o problema de corte de itens irregulares em faixa e o problema de corte de itens irregulares em placas. O diferencial do método está no decodificador que utiliza a técnica de alocação das peças combinando os cantos inferior esquerdo e superior esquerdo do recipiente através de uma heurística baseada na BL. O vetor de chaves aleatórias codifica a ordem em que os itens devem ser alocados. A estratégia para evitar a sobreposição utiliza uma malha de pontos, baseada em Toledo *et al.* (2013).

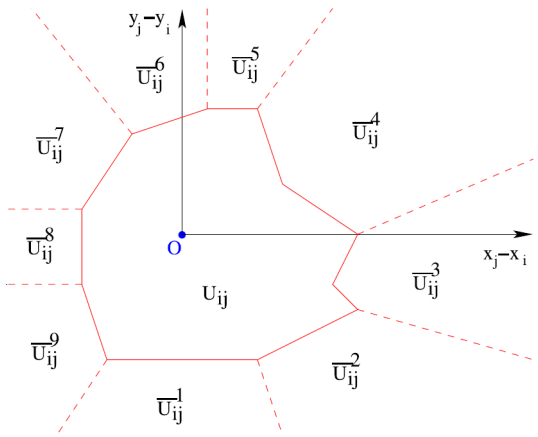
Em Mundim, Andretta e Queiroz (2017), o BRKGA é utilizado para determinar a ordem em que os itens devem ser alocados. Determinada a sequencia dos itens, os autores aplicam duas heurísticas, sendo uma baseada na BL e a outra no *no-fit raster*, para organizar os itens dentro da placa. Além disso, também apresentam o primeiro modelo quadrático inteiro para o problema com duas dimensões abertas. Por fim, em Júnior, Pinheiro e Coelho (2017), um BRKGA paralelo com múltiplas populações é proposto. Em seu método, o vetor de chaves aleatórias codifica a sequência e as rotações das peças e o decodificador é uma heurística baseada na região de colisão livre.

No caso dos métodos exatos, Carravilla, Ribeiro e Oliveira (2003) propuseram a primeira abordagem para a resolução do problema, utilizando programação por restrições. Segundo os autores, a programação por restrições fornece uma estrutura muito conveniente para esta classe de problemas, pois um empacotamento aceitável é naturalmente expresso através das restrições impostas a cada par de peças e o critério de minimização é facilmente transformado em uma estratégia de pesquisa. As restrições geométricas de não sobreposição foram construídas utilizando o NFP de cada par de peças.

Gomes e Oliveira (2006) apresentaram um algoritmo híbrido, em que a meta-heurística *Simulated Annealing* é utilizada para orientar a pesquisa sobre o espaço de solução e dois modelos de programação linear geram vizinhos durante o processo de busca. Modelos de programação linear foram escritos para representar a compactação e a separação das peças em uma dada solução, ou seja, uma vez definida a posição relativa entre as peças é escrito um modelo linear para definir o melhor posicionamento de cada peça, compactando a solução ou separando as peças sobrepostas, buscando sempre minimizar o comprimento da placa utilizado. Para evitar a sobreposição entre as peças e para alocar as peças no interior da placa os autores também utilizaram o NFP e IFP.

No trabalho de Fischetti e Luzzi (2009) foi introduzido o conceito de regiões de não sobreposição entre duas peças. Os autores particionam a região exterior do NFP, ou seja, a região na qual a segunda peça pode ser colocada sem que haja sobreposição com a primeira, em fatias (regiões) convexas. Na Figura 6, um exemplo de divisão da região exterior ao NFP é apresentado. A formulação proposta por Fischetti e Luzzi (2009) difere do modelo de Gomes e Oliveira (2006) na definição das variáveis de decisão utilizando o NFP, que permite resolver o problema como

Figura 6 – Divisão do complementar do *no-fit polygon* U_{ij} em regiões convexas.



Fonte: Fischetti e Luzzi (2009).

um todo, sem requerer a definição da posição relativa entre as peças. Além disso, o modelo inteiro misto resultante permite definir tanto a posição relativa das peças quanto seu melhor posicionamento, ou seja, permite obter a solução ótima do problema.

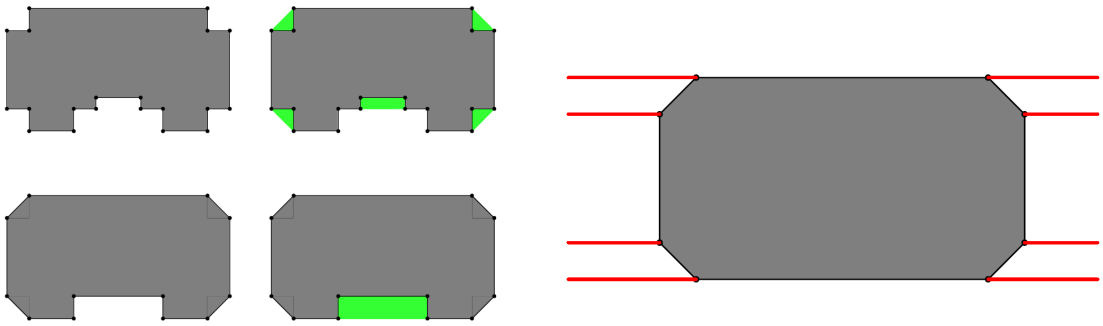
Com base nas ideias de Fischetti e Luzzi (2009), Álvarez-Valdés, Martinez e Tamarit (2013) propuseram dois modelos, chamados de HS1 e HS2, para a alocação de peças irregulares não necessariamente convexas e sem rotação, além de um método exato baseado no algoritmo de *Branch & Bound*. No modelo HS1, os autores aplicaram a técnica de *lifting* nas restrições de não sobreposição das peças, enquanto que no modelo HS2 esta técnica também foi aplicada nas restrições de alocação das peças no interior da placa. Diferentemente de Fischetti e Luzzi (2009), que não especificam a maneira em que as faixas são definidas, os autores propõem os seguintes passos para obter as regiões de não sobreposição entre duas peças: i) considerar como regiões os elementos especiais do NFP (ponto, linha e regiões interiores); ii) retirar as concavidades, isto é, deixar o NFP convexo, conforme ilustrado na Figura 7a; e iii) definir faixas horizontais para as arestas do NFP, como pode ser visto na Figura 7b.

Também em 2013, Toledo *et al.* (2013) apresentaram um modelo matemático inteiro misto, onde a placa é representada por uma malha de pontos cuja distância horizontal e vertical entre os pontos é fixa e dada por g_x e g_y, respectivamente. Na Figura 8 um exemplo de discretização da placa por pontos é apresentado.

No modelo, as variáveis binárias de decisão estão associadas à alocação de um tipo de peça em um ponto específico da placa. Note que este é um ponto positivo deste modelo, uma vez que as variáveis de decisão estão relacionadas aos tipos de peças e não às suas demandas. Da mesma forma, o tratamento geométrico também está condicionado à malha de pontos. Na Figura 9 é apresentado um IFP condicionado a malha de pontos.

Os autores destacam que a precisão e a eficiência deste método estão diretamente ligadas

Figura 7 – Estratégia para definir as regiões de não sobreposição.



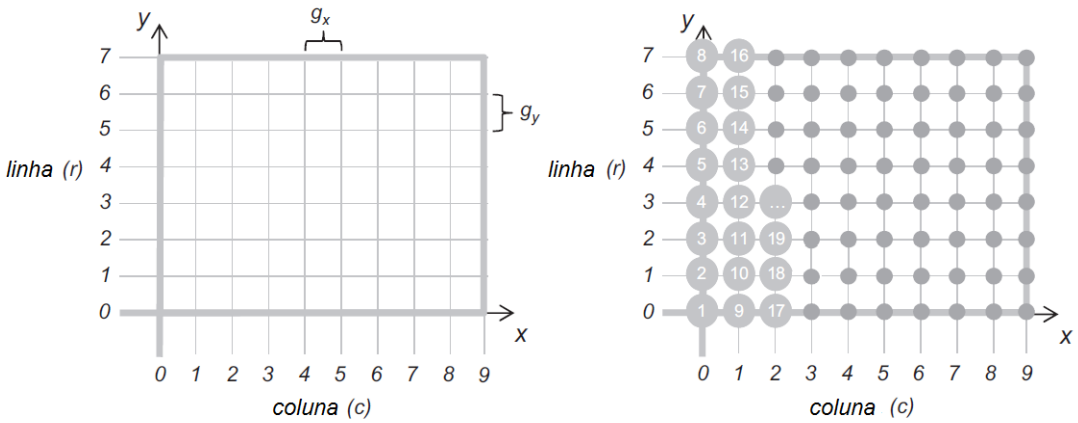
(a) Transformação do NFP em um polígono convexo.

(b) Faixas horizontais.

Fonte: Álvarez-Valdés, Martinez e Tamarit (2013).

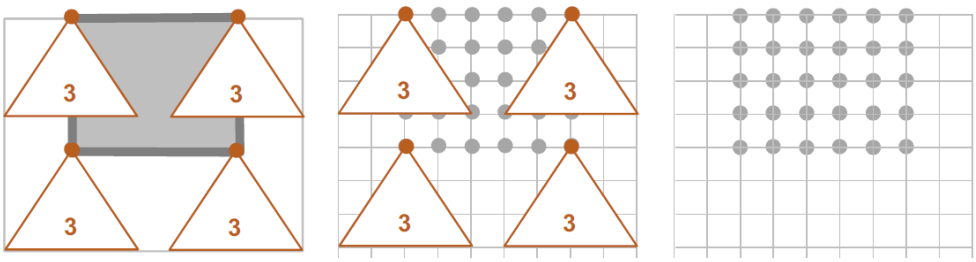
.

Figura 8 – Representação da placa por uma malha de pontos.



Fonte: Adaptada de Toledo *et al.* (2013).

Figura 9 – IFP dado uma malha de pontos.



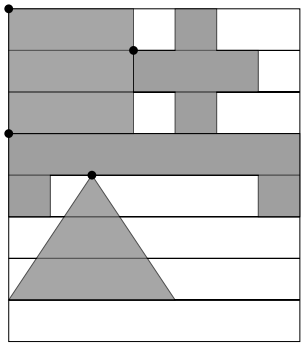
Fonte: Toledo *et al.* (2013).

à resolução da malha utilizada para representar a placa e as soluções encontradas pelo modelo podem não ser as melhores soluções para o problema de empacotamento original sobre domínios contínuos. A grande vantagem do modelo é a sua insensibilidade à geometria da peça, tornando

mais fácil estendê-lo a problemas mais complexos, tais como utilização de placas não convexas e com defeitos.

Leão *et al.* (2016) propuseram o primeiro modelo semi-contínuo, em que tanto a placa quanto o NFP e o IFP são discretizados por um sistema de coordenadas (x, y), em que x assume valores contínuos e y valores discretos. Assim, a placa é discretizada por faixas e os pontos de referência das peças podem ser alocados apenas sobre as bordas das faixas, como é ilustrado na Figura 10.

Figura 10 – Empacotamento e placa discretizada por faixas.



Fonte: Leão *et al.* (2016).

Embora Toledo *et al.* (2013) tenham conseguido resolver instâncias com muitas peças, o modelo proposto utiliza um grande número de variáveis binárias e de restrições necessárias para representar a não sobreposição das peças em cada ponto da malha. Assim, Rodrigues e Toledo (2017) propõem melhorias ao modelo de Toledo *et al.* (2013), utilizando o conceito de cobertura por cliques para reduzir o número de restrições e melhorar sua relaxação. No novo modelo, as restrições que definem o comprimento utilizado da placa são substituídas por um conjunto baseado na cobertura por cliques e as restrições que garantem a não sobreposição entre as peças são substituídas por uma cobertura de arestas por clique.

Também visando melhorar o modelo de Toledo *et al.* (2013), Cherri *et al.* (2018) propõem a estratégia de criar malhas de pontos específicas para cada peça, baseadas em sua geometria, além de apresentar uma estrutura de dados capaz de representar tais malhas. Note que nessas malhas irregulares as distâncias entre os pontos da placa não precisam ser necessariamente iguais entre si, o que adiciona uma maior flexibilidade na alocação das peças. Segundo os autores, com isso é possível superar a perda de precisão originada pela discretização da placa, uma vez que as soluções do empacotamento estão condicionadas à malha de pontos utilizada. Por fim, em Cherri *et al.* (2016), os autores propõem dois modelos inteiros, um usando trigonometria direta e outro usando o NFP. Ambos os modelos consideram peças não convexas como um conjunto decomposto de partes convexas.

CAPÍTULO

3

O PROBLEMA DE DETERMINAÇÃO DO CAMINHO DE CORTE

O problema de determinação do caminho de corte (CPDP - do inglês *Cutting Path Determination Problem*) consiste em encontrar a melhor trajetória que uma ou mais ferramentas de corte devem percorrer para executar um dado empacotamento. Uma investigação detalhada da literatura mostrou que existem diversas características que influenciam na determinação de um caminho de corte. Algumas destas características são detalhadas nas próximas seções. Na Seção 3.1 são apresentadas informações sobre o deslocamento, o material e a quantidade de ferramentas de corte disponíveis durante o processo de corte, na Seção 3.2 são apresentadas as características referentes à placa, como materiais e espessura, e na Seção 3.3 são abordados os diferentes tipos de corte. Por fim, uma revisão sobre os modelos matemáticos e métodos aplicados ao CPDP é apresentada na Seção 3.4.

3.1 Ferramentas de corte

Uma máquina de corte pode possuir uma ou mais ferramentas (cabeças) de corte, o que influencia diretamente na estratégia de empacotamento e de corte das peças. O caso mais simples é a existência de apenas uma ferramenta, que é responsável por cortar todas as peças. Nesse caso, há uma maior flexibilidade para o empacotamento das peças, logo, em geral tem-se um melhor aproveitamento da placa durante o empacotamento, como ilustrado na Figura 12a.

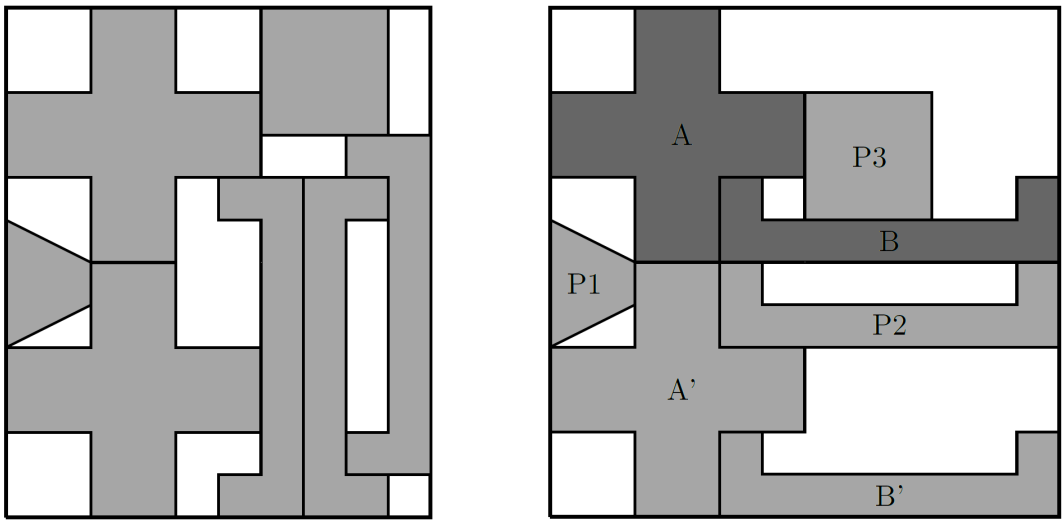
No caso mais geral, em uma máquina com múltiplas cabeças, todas as ferramentas estão fixas a um mesmo eixo, que se movimenta ao longo da placa durante o processo de corte, conforme ilustrado na Figura 11. Nesse caso, uma das cabeças de corte é considerada a mestre, enquanto que as demais seguem seus movimentos. Isso faz com que seja necessária uma mudança no empacotamento das peças, que devem respeitar as restrições de mobilidade das ferramentas, gerando padrões semelhantes a serem executados, simultaneamente, pelas cabeças de corte,

como apresentado na Figura 12b, em que as peças A, A', B e B' são cortadas simultaneamente por duas cabeças de corte distintas.

Figura 11 – Máquina de corte com múltiplas cabeças.

Fonte: Adaptada de http://oxipira.com.br/serie-master-padrao-oxicorte.php (Acesso em 01/02/2015 às 17:04).

Figura 12 – Exemplo de empacotamento.



(a) Para uma cabeça de corte. (b) Para duas cabeças de corte.

Fonte: Elaborada pela autora.

Note que é possível que existam peças que não estão presentes no caminho de corte de todas as ferramentas, como é o caso das peças P1 a P3 na Figura 12b. Nesse caso, elas são cortadas pela ferramenta mestre, enquanto as demais cabeças são automaticamente desligadas. Isso é necessário para evitar que alguma ferramenta, seguindo o movimento da mestre, danifique uma ou mais peças.

Cada máquina de corte possui um conjunto de movimentos que podem ser utilizados pelas ferramentas durante o processo de corte, dentre eles o movimento aéreo (a cima da placa). Este é um recurso muito utilizado nos casos em que as peças possuem buracos, como em Dewil, Vansteenwegen e Cattrysse (2011), Dewil, Vansteenwegen e Cattrysse (2014), ou quando o

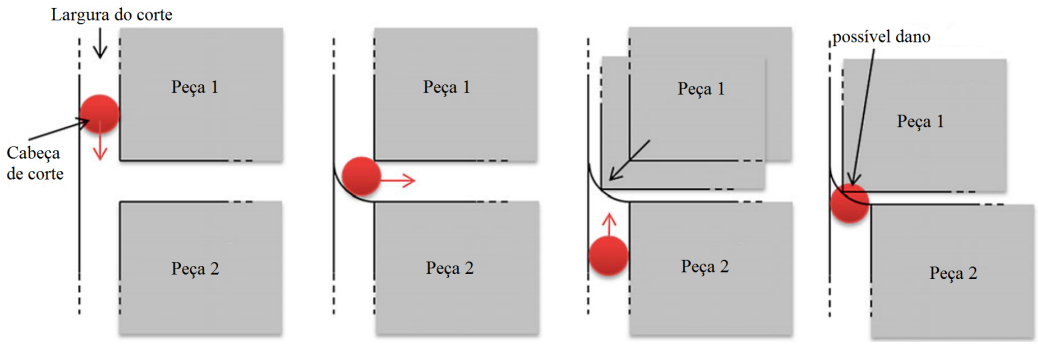
problema de aquecimento da placa é relevante, como em Han e Na (1999). Porém, isto resulta em um acréscimo no tempo total do processo, causado pela articulação vertical da cabeça de corte para cima e para baixo.

Há alguns casos, como em Rodrigues e Ferreira (2011) e Moreira *et al.* (2007), que a ferramenta não pode executar o movimento aéreo, ou seja, todo movimento da ferramenta resulta em um corte na placa. Vale ressaltar, que nestes casos, a ferramenta não pode ser deslocada pelo interior das peças, sob pena de danificá-las.

Outra característica importante sobre a ferramenta de corte é o material utilizado para o corte, que pode ser plasma, laser, jato de água, oxicorte, entre outros. Cada um destes materiais possue um conjunto específico de restrições, por exemplo, Han e Na (1999) estudam o problema de minimizar o caminho de corte feito por uma ferramenta a laser minimizando também o aquecimento causado na placa. Outro exemplo é a utilização do diamante artificial como ferramenta de corte, que possui capacidade de corte limitada, associada a sua vida útil e ao desgaste sofrido durante o processo de corte.

Em alguns casos, a ferramenta de corte pode possuir um diâmetro considerável, conforme estudado por Imahori *et al.* (2008) e Dewil, Vansteenwegen e Cattrysse (2011), Dewil, Vansteenwegen e Cattrysse (2014), e isso também influencia na determinação do empacotamento das peças, pois este deve permitir que exista um espaçamento suficiente entre as peças que será desgastado pela ferramenta durante o processo de corte. Nesse caso, deve ser considerada a movimentação que a peça já cortada pode sofrer dentro da placa, conforme ilustrado na Figura 13. Caso a peça em questão possua buracos, ou seja, necessite de cortes em seu interior, eles devem ser feitos antes da peça ser inteiramente cortada evitando possíveis danos. Essa característica é abordada em Dewil, Vansteenwegen e Cattrysse (2011), Dewil, Vansteenwegen e Cattrysse (2014).

Figura 13 – Ferramenta de corte com diâmetro.



Fonte: Adaptada de Dewil, Vansteenwegen e Cattrysse (2014).

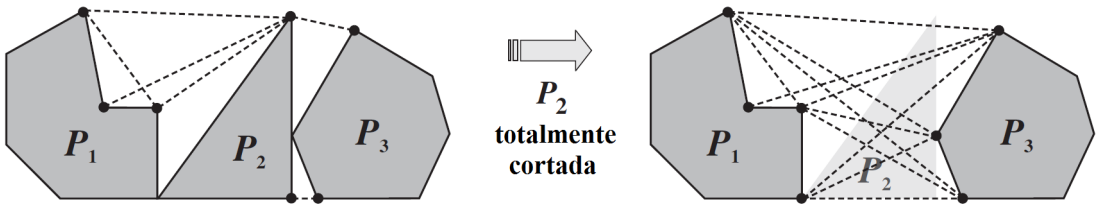
Em resumo, com relação às ferramentas de corte, quatro características foram descritas: i) quantidade de cabeças de corte; ii) tipos de movimentos a elas permitidos; iii) materiais utilizados para o corte; iv) e a possibilidade da ferramenta possuir um diâmetro não desprezível.

3.2 Placas

Dentre os diversos materiais das placas, destacam-se o metal, o couro, o tecido, a rocha e o plástico. O material da placa influencia não apenas no empacotamento, mas também em como ele é cortado. Existem materiais que demandam cortes em direções específicas, por razões relativas à estrutura própria do material, que de outra forma sofreriam danos ou desgastes inadequados. Outra importante característica da placa é sua espessura. As placas podem ter uma espessura fina ou grossa, dependendo do seu material.

A altura em que a placa é posicionada no momento do corte e os movimentos que podem ser executados pela ferramenta estão diretamente relacionados. Nesse caso, existem duas possíveis situações: a placa está suspensa, assim as peças cortadas caem e se afastam da ferramenta; ou a placa não está necessariamente suspensa e as peças cortadas não se afastam da placa. No primeiro caso, estudado por Moreira *et al.* (2007), o fato da peça se distanciar da ferramenta permite que esta se movimente pelo espaço antes preenchido pela peça, conforme ilustrado na Figura 14.

Figura 14 – Movimentos possíveis após o corte de P_2.



Fonte: Adaptada de Moreira *et al.* (2007).

No segundo caso, o mais estudado na literatura, a peça cortada permanece próxima da placa obrigando a ferramenta a tratá-la como uma peça ainda não cortada e possibilitando apenas o movimento aéreo na região ocupada pela peça.

Outra situação inerente a placa é o fato dela poder vibrar e/ou entortar durante o processo de corte. Isso pode acontecer por vários motivos, dentre eles o superaquecimento, citado em Manber e Israni (1984) e Dewil, Vansteenwegen e Cattrysse (2011), muito comum em placas de espessura fina, ou o sobrepeso em algumas regiões da placa, como estudado em Anand e Babu (2015). Imahori *et al.* (2008) também adicionaram em seu trabalho restrições que evitavam situações de vibração e deformação da placa.

Em resumo, com relação às placas, quatro características foram descritas: i) os diferentes materiais; ii) a espessura, fortemente relacionada ao material da placa; iii) o posicionamento da placa no momento do corte; iv) e as situações que podem danificar as peças.

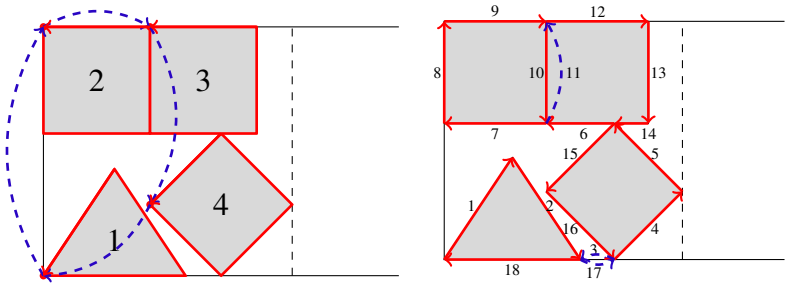
3.3 Tipos de cortes

Os cortes podem ser divididos em: furos (do inglês *piercing*), cortes não efetivos, cortes efetivos, e pré-cortes. A perfuração, um tipo especial de corte, é frequentemente o corte a ser executado no início, ou na retomada, de um caminho de corte. Após um deslocamento a cima da placa (movimento aéreo) a ferramenta inicia o corte e, sem se movimentar, realiza um furo na placa. Segundo Dewil, Vansteenwegen e Cattrysse (2014), um processo de furo pode ter seu tempo desprezado caso a placa cortada possua espessura fina, menor que 6mm. Em Manber e Israni (1984) o objetivo é minimizar o número de furos necessários durante o corte das peças.

Os cortes não efetivos, também chamados de pontes, são aqueles executados em qualquer parte da placa não relacionada a uma peça, ou seja, na sucata. Frequentemente, esse corte é utilizado para ligar peças vizinhas, cujo custo de corte da placa é menor que o custo de locomoção a cima da placa somado ao custo de perfuração.

Já os cortes efetivos correspondem aos cortes que originam as bordas das peças. Do ponto de vista geométrico, esses cortes podem ser divididos em dois casos: os cortes efetivos por perímetro, chamados de corte por peça (Figura 15a), e o corte por arestas (Figura 15b). No primeiro caso, estudado em Lee e Kwon (2006), uma vez iniciado o corte de uma peça, ela é cortada em sua totalidade antes da ferramenta de corte seguir para a próxima peça. Enquanto no segundo caso, o mais comum na literatura, os cortes não consideram a peça como um todo, cortando arestas independentes e sem uma sequência específica, como em Rodrigues e Ferreira (2011).

Figura 15 – Estratégias de caminho de corte.



(a) Peça. (b) Aresta.

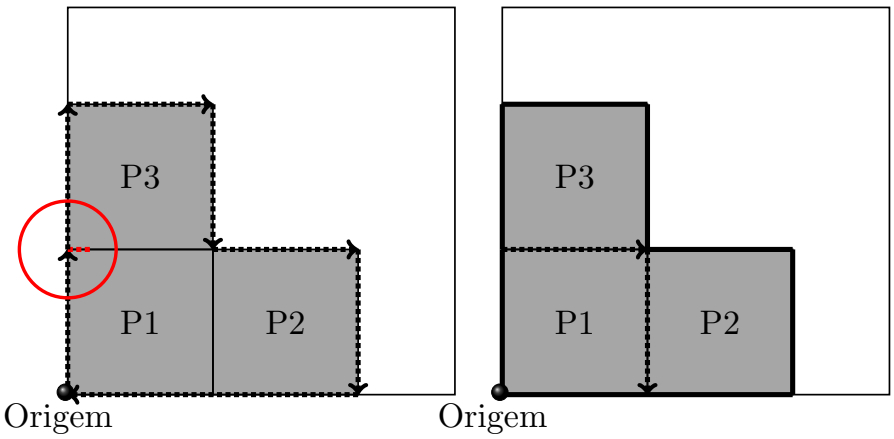
Fonte: Elaborada pela autora.

Outro corte considerado como efetivo por arestas é o corte em comum. Ele acontece quando duas ou mais arestas estão perto o suficiente para serem cortadas de uma única vez. Isso faz com que o tempo e o caminho de corte sejam reduzidos. Em ambos os trabalhos, Dewil, Vansteenwegen e Cattrysse (2011) e Dewil, Vansteenwegen e Cattrysse (2014), os autores consideram a possibilidade do corte em comum.

Por fim, dependendo da composição da placa, conforme citado em Dewil, Vansteenwegen

e Cattrysse (2014), a perfuração pode se tornar um processo custoso. Uma alternativa nessa situação é realizar um pré-corte, que consiste em fazer um pequeno corte na aresta ainda não cortada, ou na sucata próxima a ela, de forma que seja possível utilizar este ponto como entrada da ferramenta para a retomada do corte num momento futuro. Na Figura 16 é ilustrado um caminho de corte com a utilização do pré-corte entre as peças P1 e P3.

Figura 16 – Caminho de corte com utilização de pré-corte.



(a) Primeira etapa de corte. (b) Segunda etapa de corte.

Fonte: Elaborada pela autora.

Em resumo, os tipos de cortes podem ser divididos em: i) furos; ii) cortes não-efetivos; iii) cortes efetivos; iv) e pré-cortes.

3.4 Revisão da literatura

É possível encontrar na literatura diversas estratégias para resolver o CPDP. No primeiro trabalho encontrado sobre o tema, Manber e Israni (1984) propõem um algoritmo cujo objetivo é encontrar o caminho de corte com menor número de furos, impondo que cada aresta das peças (arestas obrigatórias) fosse cortada exatamente uma vez. As demais arestas, ou seja, aquelas que não pertencem a nenhuma peça (arestas não obrigatórias), podem ou não ser cortadas. Dada as condições tecnológicas da época, o algoritmo proposto necessitava da intervenção humana na criação das arestas não obrigatórias que passassem pela sucata da placa.

Han e Na (1999) tratam o problema considerando dois objetivos: minimizar o comprimento do caminho de corte e minimizar o efeito de aquecimento da placa causado pelo corte. O problema é abordado como um caixeiro viajante (TSP - do inglês *Traveling Salesman Problem*) e resolvido a partir de um algoritmo baseado em *Simulated Annealing*, que possui a minimização do efeito de aquecimento incorporado à função de custo. Nos testes computacionais

foram utilizados exemplos reais e os resultados permitiram aos autores concluírem que o método proposto é tanto eficiente quanto eficaz para resolver problemas de otimização de grande escala.

Em Lee e Kwon (2006), o objetivo é determinar o caminho de corte minimizando a distância total ociosa, ou seja, a distância percorrida pela ferramenta de corte sem cortar nada. O problema é formulado como um carteiro chinês, mas é resolvido através do problema de caixeiro viajante generalizado (GTSP - do inglês *Generalized Traveling Salesman Problem*). O grafo é formado por m vértices, sendo o primeiro relacionado ao depósito, onde o percurso deve ser iniciado e finalizado, e os demais referentes aos pontos de referência das peças. Logo, uma vez iniciado o corte de uma peça, ela deve ser inteiramente cortada, como é ilustrado na Figura 15a. Note que o ponto de referência pode estar em qualquer parte do contorno da peça, e não necessariamente em um vértice. Os autores propõem um algoritmo genético recursivo de dois passos, sendo o primeiro responsável por localizar os melhores locais para os pontos de referência e o segundo pelo sequenciamento do corte das peças.

O principal diferencial do trabalho de Moreira *et al.* (2007) é o fato da superfície a ser cortada estar suspensa, de forma que as peças cortadas se separam da placa. Os autores optaram por modelar esta característica utilizando o problema do carteiro rural dinâmico, cujo grafo é dinamicamente alterado toda vez que uma peça é inteiramente cortada. Novas arestas são criadas entre vértices que anteriormente não se conectavam, conforme ilustrado na Figura 14. O objetivo do problema é cortar todas as peças no menor tempo, sem atravessar o interior das peças que ainda não foram totalmente cortadas. Testes foram realizados utilizando exemplos de problemas reais de indústrias de cortes de metal.

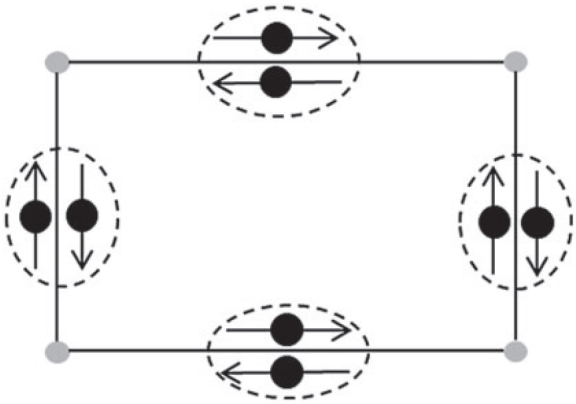
Rodrigues e Ferreira (2011) abordam o mesmo problema que Moreira *et al.* (2007), porém sem considerar o distanciamento entre as peças cortadas e a placa. Desta forma, modelam o problema como um carteiro rural e propõem o primeiro método baseado em algoritmo memético para este tipo de problema. O objetivo é minimizar a distância total percorrida, ou seja, minimizar a utilização dos cortes não efetivos. Os testes foram realizados com instâncias da literatura.

Em Imahori *et al.* (2008), o objetivo é apresentar um método heurístico para encontrar o menor caminho de corte, uma vez que o corte é muito lento, pois a placa é de um material muito rígido e a cabeça de corte possui um diamante sintético. O diferencial do trabalho é incorporar restrições práticas ao problema, como a cabeça de corte possuir um diâmetro significativo. Segundo os autores, isto acarreta a necessidade de aparar possíveis rebarbas deixadas nas peças pela máquina de corte. Outra característica real do problema considerado pelos autores é a necessidade de evitar vibração e deformação na placa.

Dewil, Vansteenwegen e Cattrysse (2011) estudam o problema de corte de peças em uma superfície de metal, com o diâmetro da ferramenta de corte relevante, a possibilidade de alocação de peças no interior de outras peças e o corte em comum das arestas. O corte das peças é feito por arestas e para a modelagem matemática é utilizado o GTSP, em que cada aresta está relacionada a um distrito que possui duas cidades, uma para cada direção de corte da aresta, conforme ilustrado

na Figura 17. O objetivo do problema é minimizar a distância total percorrida pela cabeça de

Figura 17 – Representação dos distritos de um GTSP correspondentes ao contorno de uma peça retangular.



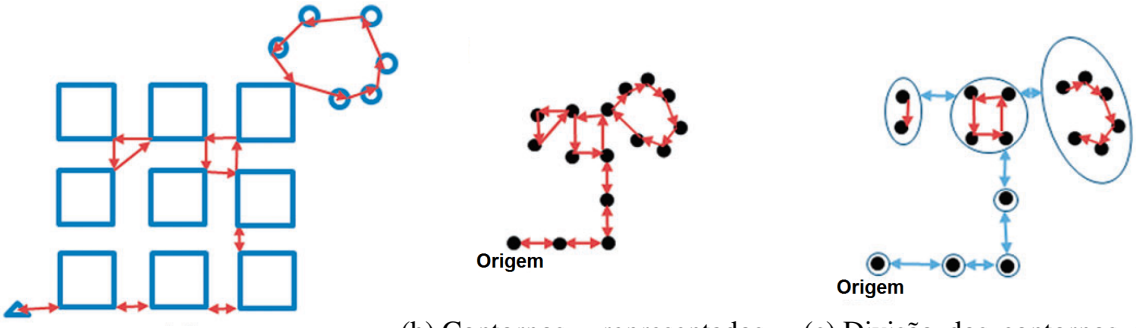
Fonte: Dewil, Vansteenwegen e Cattrysse (2011).

corte. Os autores propuseram um modelo matemático, mas devido à complexidade do problema e ao alto tempo computacional necessário para resolvê-lo, foram propostas heurísticas construtivas, de melhoramento e uma Busca Tabu para tratar o problema.

Em Dewil, Vansteenwegen e Cattrysse (2014), os autores adicionam custos aos furos, no problema anteriormente estudado, e consideram a estratégia de pré-corte como uma alternativa para evitar estes custos adicionais de perfuração. Seu objetivo é minimizar o tempo total necessário para cortar todas as peças na placa de metal levando em conta o custo (tempo) de perfuração, de pré-corte e do movimento aéreo. Como no trabalho anterior, os autores também propuseram um modelo matemático baseado no GSTP, mas devido à sua complexidade, optaram por resolver o problema utilizando diversas heurísticas construtivas.

Em Dewil *et al.* (2015), os autores propõem uma nova maneira de representar o problema estudado Dewil, Vansteenwegen e Cattrysse (2014). Segundo os mesmos, é muito provável que o caminho de corte para uma instância com peças grandes e pequenas, como ilustrado na Figura 18a, possua uma parte que se assemelhe a uma árvore e uma parte que se assemelhe a um GTSP. A nova abordagem busca encontrar uma partição para os contornos, como ilustrado na Figura 18c, em que a estrutura formada entre os grupos é modelada com uma arborescência e um GTSP é resolvido dentro de cada grupo, visando minimizar a distância total percorrida. O método proposto foi capaz de obter caminhos de corte, em média, 4,2% melhores que a heurística construtiva, descrita em Dewil, Vansteenwegen e Cattrysse (2014), e com uma melhora máxima de 11,1%. Por fim, em Dewil, Vansteenwegen e Cattrysse (2015), os autores dissertam sobre algumas características práticas do problema de caminho de corte que não foram abordadas nos trabalho anteriores. Um exemplo é a utilização das bordas da placa como bordas das peças, resultando em um menor número de arestas a serem cortadas. Os autores alertam que a utilização destas bordas pode criar regiões, conforme ilustrado na Figura 19, em que, caso peças sejam ali alocadas, novas restrições de precedência devem ser consideradas.

Figura 18 – Representação do problema proposta por Dewil *et al.* (2015).



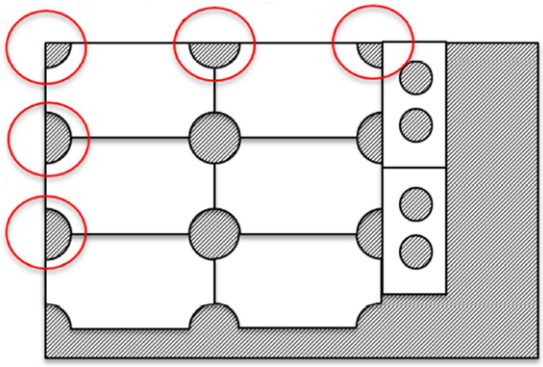
(a) Contorno das peças.

(b) Contornos representados por vértices.

(c) Divisão dos contornos em grupos.

Fonte: Adaptada de Dewil *et al.* (2015).

Figura 19 – Alocação de peças nas extremidades da placa.



Fonte: Dewil, Vansteenwegen e Cattrysse (2015).

Além disso, uma combinação de cortes não efetivos pode separar um pedaço de placa (sucata) do restante da placa. Nesse caso, também há a necessidade de considerar restrições de precedência entre as peças, pois é possível que uma outra peça esteja alocada na região cujo corte se torna inviável após a peça mais externa se separar da sucata da placa.

Os autores também estendem os trabalhos anteriores considerando o aquecimento da placa. Segundo os mesmos, os trabalhos existentes na literatura que se preocupam com essa questão consideram apenas o calor gerado localmente e não todo aquecimento causado por todos os cortes em todas as regiões da placa. Com o método proposto foi possível obter caminhos factíveis sem aquecimento, com um aumento não significativo no tempo de execução. Contudo, os autores ressaltam que o método é fortemente dependente do parâmetro de penalidade do superaquecimento.

Usberti, França e França (2011) propuseram o problema de roteamento em arcos capacitados aberto (OCARP - do inglês *Opened Capacited Arc Routing Problem*). O OCARP não considera a existência de um depósito, ou seja, não há uma restrição que obrigue a formação de

ciclos, o que determina o termo “aberto” do nome. Dentre as diversas aplicações existentes para o problema, os autores citam o CPDP. Neste caso, as cabeças de corte possuem uma quantidade de energia limitada a ser utilizada e não devem cortar partes internas das peças demandadas, para não invalidá-las. Existe a possibilidade de articulação a cima da superfície da placa, sem realizar nenhum corte, resultando no acréscimo dos tempos de articulação vertical da cabeça de corte para cima e para baixo.

Para que o problema CPDP possa ser adaptado para ao OCARP, os autores propõem alterações em suas definições, como a conversão dos vértices e das arestas das peças a serem cortadas em vértices e das arestas pertencentes ao subconjunto de arestas demandadas no OCARP; e a transformação das arestas que não pertencem às peças demandadas em dois tipos de arestas: arestas que não atravessam o interior de nenhuma peça na superfície da placa e arestas que representam deslocamentos das cabeças de corte a cima desta superfície. O custo de cada aresta corresponde ao tempo gasto para a cabeça de corte atravessá-la, enquanto que a demanda representa a energia gasta para realizar o corte na superfície e os veículos correspondem às cabeças de corte, cuja capacidade é equivalente à quantidade de energia permitida para cada cabeça de corte gastar.

Com as variáveis e as restrições adaptadas, o objetivo do problema consiste em minimizar o tempo total gasto para cortar todas as peças da superfície. Os autores propõem um modelo matemático para o OCARP, baseado no modelo do CARP proposto por Golden e Wong (1981), porém os testes realizados não utilizam instâncias de corte.

CAPÍTULO

4

O PROBLEMA INTEGRADO DE EMPACOTAMENTO E CAMINHO DE CORTE

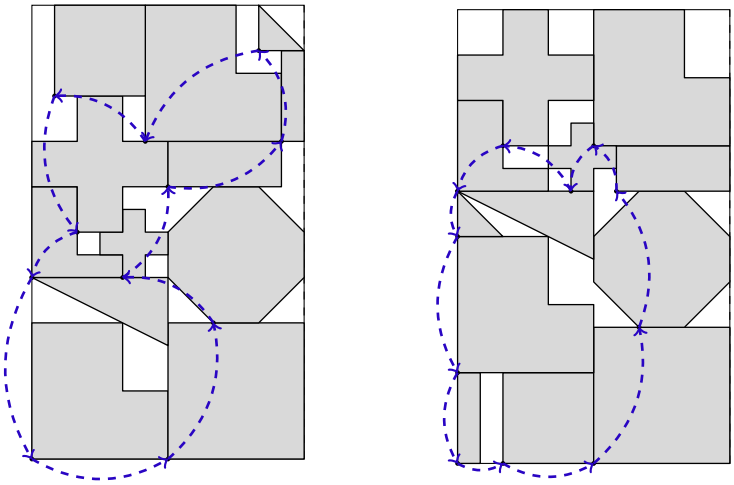
Tanto na literatura quanto na prática, os problemas de empacotamento e de caminho de corte são resolvidos de forma independente. Em geral, primeiramente se determina o posicionamento das peças que devem ser cortadas (empacotamento) e, em seguida, o caminho que a ferramenta de corte deve percorrer a fim de cortar as peças (caminho de corte). No entanto, a relevância de resolvê-los de forma integrada é destacada na literatura desde 1984. Manber e Israni (1984) afirmam que as decisões tomadas na escolha de um empacotamento influencia fortemente a qualidade da solução do problema de caminho mínimo de corte.

Na Figura 20, apresentamos um exemplo, em que na Figura 20a temos um empacotamento e caminho de corte obtidos através da estratégia hierárquica, em que primeiramente determinamos o empacotamento das peças e depois o caminho de corte, enquanto que na Figura 20b o empacotamento e o caminho de corte são obtidos junto, a partir de uma estratégia integrada. Note que, em ambos os casos, os empacotamentos possuem o mesmo comprimento contudo, a estratégia integrada conseguiu um caminho de corte (movimentos ociosos) menor.

Tanto o problema de empacotamento quanto o problema de determinação do caminho de corte são problemas de difícil resolução e, pelo o que é de conhecimento da autora, apenas três trabalhos se dedicaram a estudá-los, são eles: Pott e Glaab (2003), Sherif, Jawahar e Balamurali (2014) e Anand e Babu (2015). Os dois primeiros abordam a resolução dos problemas de forma sequencial, ou seja, primeiro determinam o empacotamento das peças e, em seguida, o caminho que a ferramenta de corte deve seguir. Enquanto que Anand e Babu (2015) são os únicos que resolvem os problemas de forma verdadeiramente integrada. Na Tabela 1, estão resumidas as principais características desses três artigos.

Neste capítulo, estes três artigos são detalhados. Na Seção 4.1, descrevemos os dois trabalhos que tratam os problemas utilizando a estratégia sequencial. Em seguida, na Seção 4.2 apresentamos o único trabalho que trata os problemas de forma integrada. Finalmente, na

Figura 20 – Empacotamento e movimentos ociosos do caminho de corte.



(a) Estratégia hierárquica. (b) Estratégia integrada.

Fonte: Elaborada pela autora.

Tabela 1 – Estado da arte da integração dos problemas de empacotamento e caminho de corte.

	Estratégia	Método de resolução	Tipo de corte	Empacotamento	Tipo de peças
Pott e Glaab (2003)	Sequencial	*Branch&Cut*	Arestas	Manual	Não cita
Sherif, Jawahar e Balamurali (2014)	Sequencial	Simulated Annealing	Arestas	*Software* comercial	Irregulares (envoltória quadrática)
Anand e Babu (2015)	Integrada	Algoritmo Genético	Arestas	*Bottom-Left*	Retângulos

Seção 4.3, analisamos os trabalhos da literatura e apresentamos considerações sobre a resolução de problemas de empacotamento e de caminho de corte de forma integrada.

4.1 Estratégia hierárquica

O primeiro trabalho que busca resolver os problemas de empacotamento e de caminho de corte foi publicado apenas no início dos anos 2000. Fortemente baseado em um caso real, o trabalho de Pott e Glaab (2003) tinha por objetivo propor uma estratégia sequencial de resolução dos problemas de empacotamento e de caminho de corte. O trabalho foi desenvolvido em colaboração com uma empresa que fabrica produtos de couro.

O couro, sendo um produto natural, não apresenta a mesma qualidade e cor em toda a sua

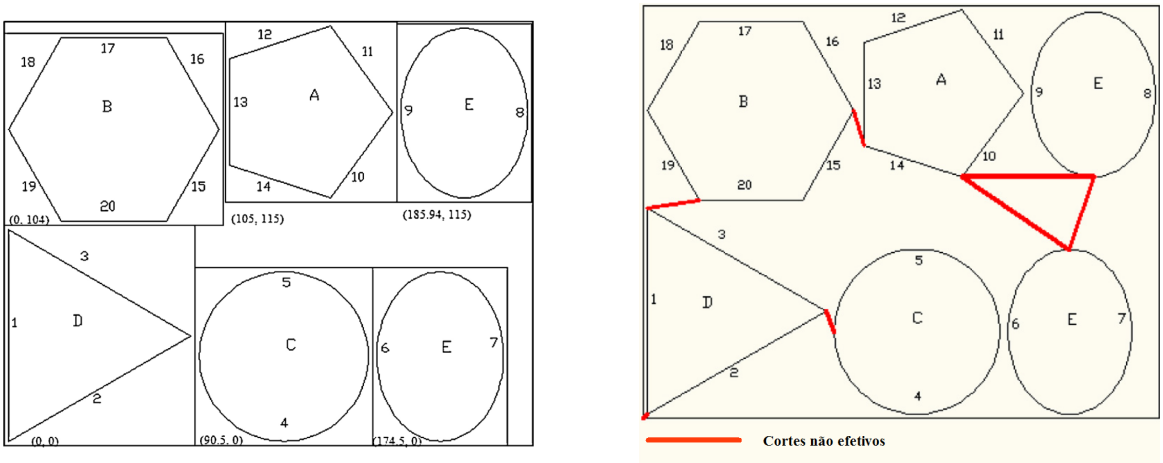
extensão, logo é importante decidir quais peças são cortadas em cada uma de suas regiões. Por exemplo, partes que não ficam visíveis no produto final podem ser cortadas a partir de regiões do couro de menor qualidade. Neste sentido, a empresa colaboradora desenvolveu uma máquina que auxilia seus funcionários durante o planejamento do empacotamento das peças, que é feito manualmente. A máquina projeta a imagem das peças no couro utilizando *lasers*.

Após definido o empacotamento das peças, a segunda etapa é o corte do couro, ou seja, determinar o caminho a ser percorrido pela ferramenta de corte. Esse processo é o único inteiramente automático. Nesse caso, não importava para a empresa o menor caminho de corte, mas sim a qualidade das bordas das peças que, segundo os autores, depende do caminho de corte e dos ângulos feitos pela ferramenta de corte. Os autores propõem um modelo matemático e um algoritmo *Branch & Cut* para tratar o problema. Vale destacar que o corte é planejado por aresta.

Em 2014, Sherif, Jawahar e Balamurali (2014) também estudaram os problemas de empacotamento e de caminho de corte e, assim como Pott e Glaab (2003), também optaram por uma estratégia sequencial, ou seja, trataram o empacotamento e o caminho de corte separadamente. A estratégia desenvolvida visava maximizar a utilização do material na etapa de empacotamento e minimizar o comprimento dos movimentos ociosos (não efetivos) na etapa do corte.

Para o empacotamento, os autores utilizaram um *software* comercial para posicionar os retângulos envolventes das peças. Enquanto que para o corte, que era o foco do trabalho, os autores desenvolveram um algoritmo baseado em *Simulated Annealing*. Na Figura 21 é ilustrada uma solução obtida em cada fase da estratégia de Sherif, Jawahar e Balamurali (2014). Os autores

Figura 21 – Soluções obtidas por Sherif, Jawahar e Balamurali (2014).



(a) Empacotamento ótimo.

(b) Cortes não efetivos.

Fonte: Adaptada de Sherif, Jawahar e Balamurali (2014).

destacam que um trabalho futuro relevante é estender a abordagem apresentada para resolver de forma integrada os problemas utilizando, por exemplo, uma estratégia multiobjetivo.

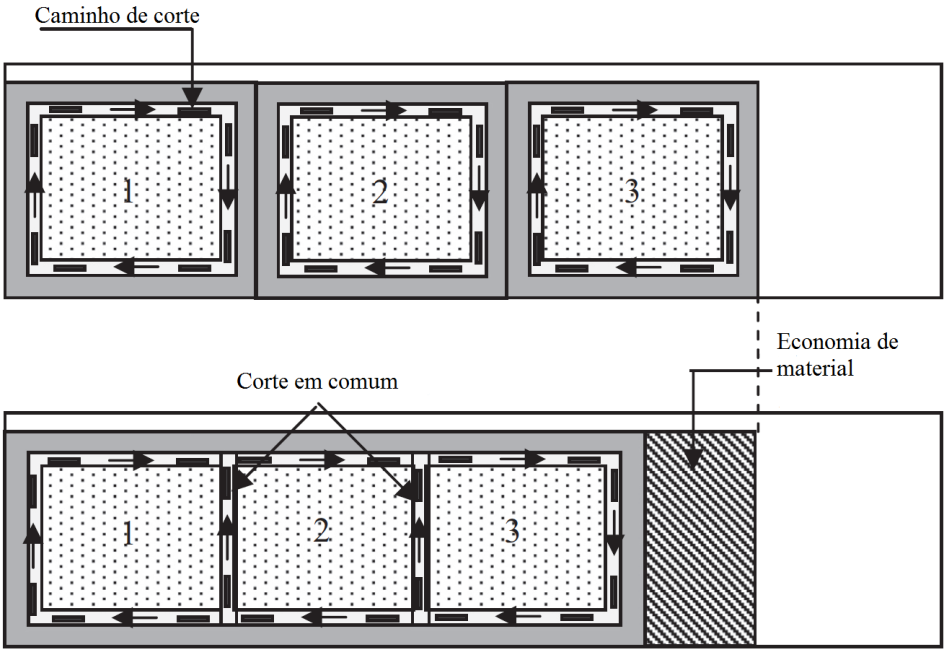
Os dois trabalhos aqui descritos foram as primeiras tentativas de analisar os problemas de forma conjunta, no entanto, os problemas ainda não são resolvidos de forma integrada.

4.2 Estratégia integrada

Anand e Babu (2015) foram os primeiros a resolver os problemas de empacotamento e de caminho de corte de corte de forma integrada. Os autores buscaram encontrar um empacotamento de peças retangulares em uma placa também retangular considerando as perspectivas de fabricação, ou seja, algumas características do processo de corte. O objetivo era minimizar a perda de material da placa e o comprimento do caminho de corte.

Uma das características do processo de corte considerada é o fato do diâmetro da ferramenta de corte não ser desprezível, logo há um desgaste da placa durante o processo de corte. Outra característica relevante é a exigência de uma distância mínima entre as peças, para que seja assegurada uma maior estabilidade da placa durante o corte, evitando, por exemplo, que a placa entorte. Devido a essas particularidades, as dimensões das peças são redefinidas antes que o empacotamento seja iniciado. Além disso, os autores também utilizam, sempre que possível, a criação de arestas em comum entre as peças, o que possibilita uma maior economia de material, conforme ilustrado na Figura 22.

Figura 22 – Vantagens do empacotamento de peças regulares em uma placa regular quando considerado o corte em comum.

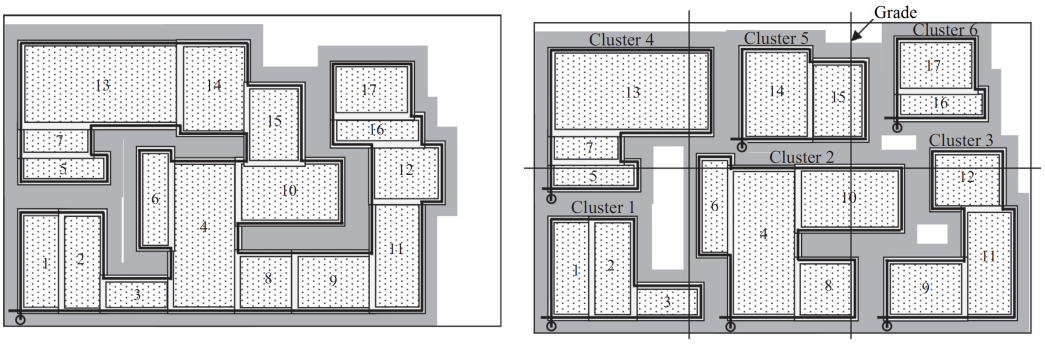


Fonte: Adaptada de Anand e Babu (2015).

Os autores desenvolveram uma heurística construtiva baseada na estratégia *Bottom-Left*. Os dados de entrada da heurística são a sequência das peças, com suas dimensões adaptadas, e

suas orientações, 0º ou 90º. Segundo os autores, a heurística apresenta uma deficiência: gera um aglomerado de peças (Figura 23a) que na prática não é viável, pois pode comprometer fisicamente a placa e as peças, uma vez que a placa pode entortar devido ao próprio peso. Para evitar isso, os autores consideraram uma estratégia de dividir a placa em áreas menores, *clusters*, de tamanho fixo, como ilustrado na Figura 23b.

Figura 23 – Disposição das peças retangulares na placa.



(a) Em um único *clusters*.

(b) Em vários *clusters*.

Fonte: Adaptada de Anand e Babu (2015).

A heurística construtiva *Bottom-Left* gera um empacotamento para cada sequência de peças e orientações. Cada combinação de sequência de peça, orientações e tamanho da grade (*clusters*) gera um empacotamento diferente. Portanto, os autores desenvolveram um algoritmo genético que tem por objetivo evoluir essas sequências, buscando a sequência que gera o empacotamento ótimo. A função de avaliação do genético proposto considera dois objetivos: a maximização da área ocupada e a minimização do comprimento do caminho de corte. Segundo os autores, a importância desses objetivos pode variar de acordo com a decisão do tomador de decisões. A estratégia sugerida é selecionar n_1 indivíduos com maior área ocupada e n_2 indivíduos com menor caminho de corte para a nova população.

Segundo os autores, o algoritmo proposto ajuda os fabricantes de componentes de chapa metálica a gerar o empacotamento com uma utilização efetiva do material de chapa, bem como a minimizar o tempo de corte por meio do arranjo das peças com arestas em comuns. Anand e Babu (2015) citam que as questões sobre o processo de corte não são levadas em conta na hora de gerar o empacotamento, mas quando consideradas a capacidade do processo de corte pode ser melhorada, minimizando o tempo de processamento e a utilização dos materiais. Além disso, embora esse trabalho estude apenas o caso com peças regulares, os autores citam que essa abordagem pode ser estendida ao problema com peças irregulares.

4.3 Considerações finais

Ambos os problemas de empacotamento de peças irregulares e de determinação do caminho mínimo de corte têm relevância tanto teórica quanto prática. Do ponto de vista prático, possuem uma grande aplicabilidade em processos industriais, que variam desde recortes em tecido até cortes de placas de metal. Já do ponto de vista acadêmico, são problemas combinatórios de difícil resolução.

Como destacado na literatura, as decisões tomadas durante o empacotamento das peças influenciam fortemente a resolução do problema de caminho mínimo de corte, porém existem poucos trabalhos que propõem estratégias que resolvam estes problemas simultaneamente. É de conhecimento da autora que, até o momento, apenas Anand e Babu (2015) tenham abordado estes problemas de forma integrada, porém sem propor um modelo matemático, apenas uma heurística que empacota e corta peças regulares. Logo, esta tese de doutorado tem como objetivo central colaborar para o preenchimento desta lacuna de pesquisa.

Parte I

Abordagens com empacotamento discreto e corte por peças

CAPÍTULO

5

MODELO INTEGRADO DE EMPACOTAMENTO DE PEÇAS IRREGULARES E CAMINHO DE CORTE POR PEÇA: VÉRTICE FIXO

Neste capítulo, é apresentado o primeiro modelo que integra os problemas de empacotamento de peças irregulares em faixa e de determinação de caminho de corte mínimo[1]. Na Seção 5.1, são apresentadas as principais características do problema estudado. Como o problema é biobjetivo, na Seção 5.2, apresenta-se uma sucinta definição sobre problemas multiobjetivos. O modelo matemático desenvolvido é descrito na Seção 5.3, enquanto os experimentos computacionais são analisados na Seção 5.4.

5.1 Características gerais do problema estudado

Com base em casos reais e em dados obtidos da literatura, consideramos as seguintes características para o problema estudado:

- ferramenta de corte:
  - possui uma única cabeça de corte, que sempre inicia seu trajeto a partir de uma origem pré-definida e retorna ao mesmo local após o corte de todas as peças;
  - pode se locomover acima da placa em movimentos aéreos;
  - possui diâmetro desprezível e a capacidade de corte é suficiente para cortar todas as peças, ou seja, a ferramenta não se desgasta durante o uso;

[1] O texto desse capítulo está fortemente baseado no artigo Oliveira *et al.* (2018), publicado na *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*.

- placa:
  - é de espessura fina, portanto o tempo de perfuração é desprezível;
  - é resistente ao calor gerado durante o processo de corte, logo nem a placa nem as peças sofrem deformações ao longo do processo de corte;
- peças:
  - continuam próximas à placa mesmo após serem cortadas;
  - não possuem buracos.

5.2 Otimização multiobjetivo

Nesta seção, apresentamos algumas definições importantes para os problemas de otimização multiobjetivo (POM). Para mais detalhes sugerimos a leitura de Deb (2014), que apresenta um completo tutorial sobre os POM. Considere o problema de otimização, em que M funções objetivo devem ser minimizadas, dado por:

$$\min \quad f_m(x) \qquad m = 1,\ldots,M, \tag{5.1}$$

$$s.a: \quad g_j(x) \leq 0, \qquad j = 1,\ldots,J, \tag{5.2}$$

$$x \in \mathbb{R}^n, \tag{5.3}$$

sendo $x = (x_1,\ldots,x_n)^T \in \mathbb{R}^n$ o vetor das variáveis de decisão e $z_m = f_m(x)$ a m-ésima função objetivo a ser minimizada, $m = 1,\ldots,M$.

Definição 1 (Espaço de decisão)**.** O espaço $X \subseteq \mathbb{R}^n$, em que $X = \{x \in \mathbb{R}^n : g_j(x) \leq 0, j = 1,\ldots,J\}$, é chamado de espaço de decisão ou espaço factível e é formado pelas J restrições $g_j(x) \leq 0, j = 1,\ldots,J$.

Definição 2 (Espaço objetivo)**.** O espaço $Z \subseteq \mathbb{R}^M$, em que $Z = \{z \in \mathbb{R}^M : z = f(x), \forall x \in X\}$, gerado pela aplicação das M funções objetivo no espaço X é chamado de espaço objetivo ou espaço de critério.

Definição 3 (Dominância)**.** Uma solução $x^1 \in X$ domina uma outra solução $x^2 \in X$, se as seguintes condições são verdadeiras:

i) a solução x^1 não é pior que a solução x^2 em todos os valores objetivos, ou seja, $f_m(x^1) \leq f_m(x^2)$, para todo $m = 1,\ldots,M$; e

ii) a solução x^1 é estritamente melhor que a solução x^2 em pelo menos um valor objetivo, ou seja, $f_m(x^1) < f_m(x^2)$, para pelo menos um $m = 1,\ldots,M$.

Se ambas as condições são satisfeitas, então dizemos que a solução x^1 domina a solução x^2 ou que a solução x^2 é dominada pela solução x^1. Note que, no caso da solução x^1 não dominar a solução x^2, isso não implica que a solução x^2 domine a solução x^1.

Definição 4 (Solução eficiente)**.** Uma solução $x^* \in X$ é denominada solução eficiente se não existir uma outra solução $x \in X$ tal que x domine x^*.

Definição 5 (Ponto não dominado)**.** Um ponto $z^* \in Z$ é chamado de ponto não dominado (ou solução ótima de Pareto) se sua imagem inversa é uma solução eficiente, ou seja, $z^* = f(x^*)$.

Definição 6 (Conjunto eficiente)**.** O conjunto eficiente, denotado por $X^* \subseteq X$, é formado por todos os elementos de X que não são dominados por nenhum elemento de X.

Definição 7 (Conjunto não dominado)**.** Também conhecido por fronteira de Pareto ou curva de Pareto, o conjunto não dominado $Z^* \subseteq Z$ é a imagem do conjunto eficiente X^*, ou seja, $Z^* = \{z^* \in Z : z^* = f(x^*), \forall x^* \in X^*\}$.

Em um POM, as funções objetivo são conflitantes entre si, ou seja, a minimização de um objetivo acarreta o aumento de pelo menos um outro objetivo. Os vetores definidos a seguir são importantes para a construção da fronteira de Pareto.

Definição 8 (Vetor ideal)**.** O vetor $z^I = (z_1^I, \ldots, z_M^I)^T \in Z$ é denominado vetor ideal quando sua m-ésima componente é o valor mínimo do subproblema mono-objetivo

$$\begin{aligned} \min \quad & z_m^I = f_m(x) \\ s.a: \quad & x \in X, \end{aligned}$$

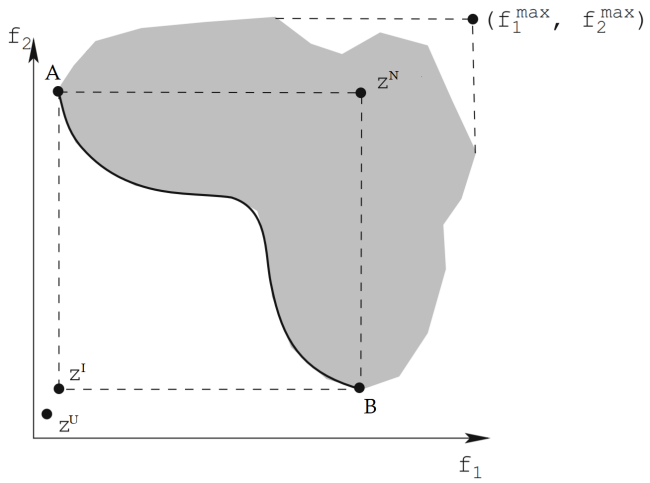
para todo $m = 1, \ldots, M$.

Definição 9 (Vetor utópico)**.** O vetor $z^U = (z_1^U, \ldots, z_M^U)^T \in Z$ é denominado vetor utópico se sua m-ésima componente é definida da seguinte maneira: $z_m^U = z_m^I - \varepsilon_m$, para todo $m = 1, \ldots, M$ e $\varepsilon_m > 0$.

Definição 10 (Vetor nadir)**.** O vetor $z^N = (z_1^N, \ldots, z_M^N)^T \in Z$ é denominado nadir se sua m-ésima componente é o maior valor da função objetivo f_m no conjunto eficiente.

O vetor nadir não deve ser confundido com o vetor de objetivos definido pelos piores valores factíveis da função f_m^{max}, conforme ilustrado na Figura 24. Os pontos z^U, z^I e z^N são, respectivamente, os vetores utópico, ideal e nadir. Os pontos A e B são pontos extremos da curva de Pareto, também conhecidos como lexicográficos e o ponto $f^{max} = (f_1^{max}, f_2^{max})$ apresenta os maiores valores para as funções $f_1(x)$ e $f_2(x)$.

Figura 24 – Pontos notáveis no espaço objetivo.



Fonte: Adaptada de Deb (2014).

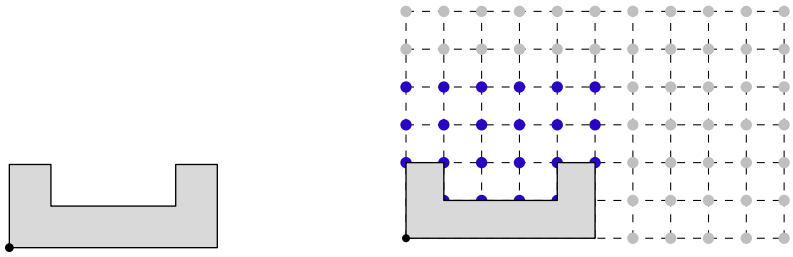
5.3 Modelo matemático

Considerando as características definidas na Seção 5.1, desenvolvemos o primeiro modelo matemático que integra os problemas de empacotamento de peças irregulares e de determinação do caminho de corte. Para o modelo, impomos que o corte é realizado por peças a partir de seu vértice de referência (vértice fixo), ou seja, o corte de uma peça é representado por um ponto pelo qual a ferramenta de corte inicia o corte da peça inteira e, após concluído, segue para o corte da próxima peça, ou retorna à origem, caso todas as peças já tenham sido cortadas. O corte por peça é preferível em algumas situações práticas, pois garante a uniformidade das propriedades físicas do material da placa e facilita a remoção das peças, que devem ser retiradas uma a uma por um trabalhador ou robô depois de cortadas.

Dois modelos da literatura foram utilizados como base para o desenvolvimento do modelo integrado: o modelo dos pontos, proposto por Toledo *et al.* (2013), que é responsável pelo empacotamento das peças e o modelo do caixeiro viajante (TSP - do inglês *Traveling Salesman Problem*), como proposto por Dantzig, Fulkerson e Johnson (1954), responsável pela determinação do caminho mínimo de corte considerando o corte por peça.

As principais vantagens de utilizar o modelo dos pontos para o empacotamento das peças são: a insensibilidade quanto à geometria das peças, uma vez que cada peça é representada por um ponto de referência, como apresentado na Figura 25a; e a capacidade de resolver instâncias com muitas peças iguais, uma vez que a variável de decisão do modelo está relacionada aos tipos de peças e não ao total de peças. Os autores representam a placa por uma malha de pontos regular em que a distância horizontal e vertical entre os pontos é dada por g_x e g_y, respectivamente, conforme ilustrado na Figura 25b. Portanto, todas as soluções obtidas estão sempre condicionadas à discretização utilizada para a placa.

Figura 25 – Exemplo de peça e placa.



(a) Peça e seu ponto de referência. (b) Discretização da placa.

Fonte: Elaborada pela autora.

Visto que no modelo dos pontos as peças são representadas por seus pontos de referência, uma proposta simples para determinar o caminho de corte é definir que as peças são cortadas a partir de seus pontos de referência e, então, determinar o menor percurso entre estes pontos. Assim, o problema de caminho de corte torna-se equivalente ao TSP, que busca encontrar o menor percurso que, saindo de uma cidade origem, visite cada cidade exatamente uma vez e retorne à cidade de origem. Para o problema estudado, as cidades são os pontos de referência das peças e a cidade origem, ou seja, o ponto em que a ferramenta de corte deve iniciar o caminho de corte e retornar ao final dele, é o ponto mais à esquerda e mais abaixo na placa, denotado por 0, de coordenadas $(0,0)$.

Vale destacar que um ponto de referência pode ser posicionado em qualquer ponto da placa, desde que respeitadas as restrições geométricas do empacotamento. Logo, temos que as restrições de continuidade do percurso do TSP estão diretamente relacionadas à utilização dos pontos da placa na alocação das peças, ou seja, um ponto da placa só poderá participar do percurso se houver pelo menos um ponto de referência de alguma peça posicionado nele.

O modelo integrado de empacotamento de peças irregulares e caminho de corte considerando o corte por peça a partir de um vértice fixo (MIEPICCP-VF) é dado por (5.4) – (5.15). Para a compreensão do modelo, considere os seguintes conjuntos, índices, parâmetros e variáveis.

Conjuntos, índices e parâmetros

- $\mathbb{T} = \{1, ..., T\}$: conjunto de tipos de peças,
- $\mathbb{C} = \{0, ..., C-1\}$: conjunto das colunas da malha de pontos,
- $\mathbb{R} = \{0, ..., R-1\}$: conjunto das linhas da malha de pontos,
- $\mathbb{D} = \{0, ..., D-1\}$: conjunto de pontos da malha ($\mathbb{D}' = \mathbb{D}/\{0\}$),
- $t, u \in \mathbb{T}$: tipos de peça,
- $c \in \mathbb{C}$: coluna da malha de pontos,

- $r \in \mathbb{R}$: linha da malha de pontos,
- $d \in \mathbb{D}$: ponto da malha de pontos, que pode ser representado por um par de valores $(c,r) \in \mathbb{C} \times \mathbb{R}$, no qual $c = \lfloor \frac{d}{R} \rfloor$ e $r = d - \lfloor \frac{d}{R} \rfloor R$. Reciprocamente, $d = cR + r$,
- q_t: número de peças do tipo t a serem cortadas,
- X_t^M: coordenada do vértice de maior valor no eixo x da peça do tipo t,
- $\mathbb{IFP}_t$: conjunto de pontos pertencentes à placa no qual o ponto de referência da peça do tipo t pode ser alocado sem que a peça extrapole os limites da placa,
- $\mathbb{NFP}_{t,u}^d$: conjunto de pontos que estão no interior do $NFP_{t,u}$ se uma peça do tipo t for posicionada no ponto d. Portanto, se t for posicionada em d e se u for posicionada num ponto pertencente a $\mathbb{NFP}_{t,u}^d$ então as peças dos tipos t e u se sobrepõem,
- T_p: total de peças ($T_p = \sum_{t \in \mathbb{T}} q_t$).

Variáveis

- z: variável real que representa o comprimento utilizado da placa,
- δ_t^d: variável binária que recebe o valor 1 se uma peça do tipo t é alocada ao ponto d da placa e 0 caso contrário,
- x_{ij}: variável binária que recebe o valor 1 se o percurso sai do ponto i da placa com destino ao ponto j da placa e 0 caso contrário,
- w_i: variável binária que recebe o valor 1 quando uma ou mais peças têm seus pontos de referência localizados no ponto i da placa e 0 caso contrário.

$$\min \quad z \tag{5.4}$$

$$\min \quad \sum_{i \in \mathbb{D}} \sum_{j \in \mathbb{D}} d_{ij} x_{ij} \tag{5.5}$$

s.a

Restrições de empacotamento

$$(c_d g_x + X_t^M)\delta_t^d \leq z, \qquad \forall d \in \mathbb{IFP}_t, \forall t \in \mathbb{T}, \tag{5.6}$$

$$\sum_{d \in \mathbb{IFP}_t} \delta_t^d = q_t, \qquad \forall t \in \mathbb{T}, \tag{5.7}$$

$$\delta_u^e + \delta_t^d \leq 1, \qquad \forall e \in \mathbb{NFP}_{t,u}^d, \forall t, u \in \mathbb{T}, u \geq t, \forall d \in \mathbb{IFP}_t, \tag{5.8}$$

Restrições de acoplamento

$$\sum_{t \in \mathbb{T}} \delta_t^i \leq T_p w_i, \qquad \forall i \in \mathbb{D}, \tag{5.9}$$

$$\sum_{i \in \mathbb{D}'} w_i \leq T_p, \tag{5.10}$$

Restrições de determinação de caminho de corte

$$\sum_{i \in \mathbb{D}, i \neq j} x_{ij} = w_j, \qquad \forall j \in \mathbb{D}', \tag{5.11}$$

$$\sum_{j \in \mathbb{D}, j \neq i} x_{ij} = w_i, \qquad \forall i \in \mathbb{D}', \tag{5.12}$$

$$\sum_{i \in \mathbb{D}'} x_{i0} = 1, \tag{5.13}$$

$$\sum_{j \in \mathbb{D}'} x_{0j} = 1, \tag{5.14}$$

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1, \qquad S \subseteq \{1, \ldots, D-1\}, \quad 1 \leq |S| \leq D-2, \tag{5.15}$$

Domínio das variáveis

$$\delta_t^d \in \{0,1\}, \qquad \forall d \in \mathbb{IFP}_t, \forall t \in \mathbb{T}, \tag{5.16}$$

$$w_i \in \{0,1\}, \qquad \forall i \in \mathbb{D}, \tag{5.17}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i,j \in \mathbb{D}, \tag{5.18}$$

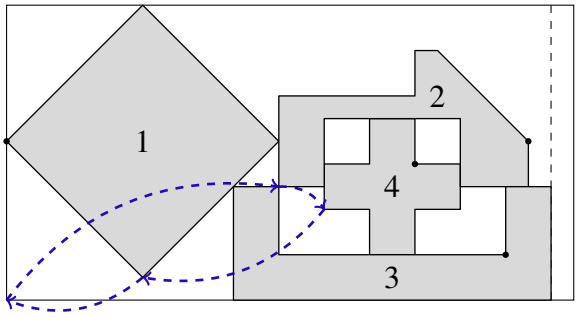
$$z \geq 0. \tag{5.19}$$

O modelo (5.4) – (5.19) visa minimizar o comprimento total da placa utilizado para a alocação das peças (5.4) e também minimizar o percurso feito pela ferramenta de corte entre as peças (5.5). As restrições (5.6) – (5.8) foram definidas como proposto em Toledo *et al.* (2013) e são responsáveis por minimizar o comprimento total da placa utilizado na alocação das peças, sendo que as restrições (5.6) definem o comprimento utilizado da placa; as restrições (5.7) garantem que todas as peças do tipo t demandadas sejam alocadas na placa; e as restrições (5.8) garantem a não sobreposição entre as peças.

As restrições (5.9) e (5.10) são responsáveis por integrar os problemas de empacotamento e de determinação de caminho de corte. As restrições (5.9) exigem que quando uma ou mais peças tenham seus pontos de referência localizados no ponto i da placa, então a variável w_i assume o valor 1, indicando que o caminho de corte deve incluir este ponto. A restrição (5.10) limita a quantidade máxima de pontos da placa associados a pontos de referências de peças, ou seja, se os pontos de referência das peças estiverem localizados em pontos distintos da placa então a somatória das variáveis w_i é igual ao total de peças (T_p), porém se alguma peça compartilhar a localização de seu ponto de referência com outra peça, respeitando as restrições de não sobreposição, então a somatória das variáveis w_i é menor que o total de peças. Na Figura 26

é ilustrada uma solução em que os pontos de referência das Peças 2 e 3 estão associados a um mesmo ponto da placa.

Figura 26 – Localização de distintos pontos de referência em um mesmo ponto da placa.



Fonte: Elaborada pela autora.

As restrições (5.11) – (5.15) definem o problema de determinação do caminho de corte como um TSP entre os pontos em que os pontos de referência das peças das peças estão alocados. As restrições (5.11) exigem que se o ponto de referência de uma peça for alocado ao ponto j ($w_j = 1$), então esse ponto deve pertencer ao caminho de corte. Logo, há uma dependência entre a utilização do ponto da placa no percurso do TSP e a localização de um ponto de referência nele, ou seja, um ponto j da placa só participa do percurso se houver pelo menos um ponto de referência de alguma peça posicionado nele ($w_j = 1$), caso contrário o ponto j não pertence ao percurso. O mesmo raciocínio é aplicado as restrições (5.12) que exigem que após uma peça ser cortada uma única outra peça é cortada.

As restrições (5.13) e (5.14), relacionadas à origem da ferramenta (ponto 0), são independentes da variável w_0, pois mesmo que um ponto de referência não esteja localizado no ponto 0, ainda assim ele deve fazer parte do percurso, uma vez que ele é o ponto de partida e de retorno da ferramenta. As restrições (5.15) são responsáveis pela eliminação de subciclos, ou seja, impõem que exista uma única rota que corte todas as peças. Por serem compostas por um número exponencial de restrições, utilizamos a estratégia de impedir os subciclos a medida que eles forem surgindo. Essa estratégia é conhecida na literatura como *lazy constraints*. Por fim, as restrições (5.16) – (5.19) definem o domínio das variáveis.

5.4 Experimentos computacionais

Os testes computacionais foram divididos em duas fases. Na primeira fase, apresentada na Subseção 5.4.1, investigamos a melhoria no caminho de corte quando o MIEPICCP-VF é utilizado para obter o empacotamento de comprimento mínimo. Na segunda fase, apresentada na Subseção 5.4.2, avaliamos o MIEPICCP-VF como um modelo biobjetivo, minimizando o comprimento do empacotamento e o caminho de corte. Em ambas as fases, os resultados foram

comparados com os obtidos através da estratégia hierárquica, em que primeiro é definido o empacotamento das peças para, em seguida, determinar um caminho de corte para as peças.

5.4.1 Fase I - Empacotamento de comprimento mínimo

Nesta primeira fase, analisamos o impacto em resolver o problema de empacotamento de peças irregulares em faixa considerando a determinação do caminho de corte como um objetivo secundário. Em outras palavras, o objetivo é avaliar se o MIEPICCP-VF, descrito na Seção 5.3, é capaz de encontrar empacotamentos com os mesmos comprimentos que os empacotamentos encontrados pelo modelo dos pontos original, porém com caminhos de corte menores. Assim, a função objetivo do MIEPICCP-VF foi reescrita como:

$$\min \qquad z + 10^{-3} \sum_{i \in \mathbb{D}} \sum_{j \in \mathbb{D}} d_{ij} x_{ij}. \tag{5.20}$$

Os experimentos computacionais foram realizados em um computador Intel Core i7-7700 (3.60GHz x 8), com 16GB de memória RAM e sistema operacional Ubuntu 16.04. Os modelos matemáticos foram implementados em linguagem C/C++ utilizando o *software* de otimização ILOG CPLEX 12.6 e foi considerado o tempo limite de 1 hora (TL = 3600s). Nesse experimento, foram utilizadas 35 instâncias da literatura do problema de corte e empacotamento, que podem ser encontradas no endereço eletrônico do ESICUP (<https://www.euro-online.org/websites/esicup/>). Detalhes sobre as instâncias, como nome, dimensões das placas, número de tipos de peças e total de peças em cada instância podem ser encontrados na Tabela 2. Devido à discretização da placa, um limitante superior para seu comprimento, grande o suficiente para que nenhuma solução seja perdida, deve ser definido. As soluções do MIEPICCP-VF são comparadas àquelas encontradas utilizando a estratégia hierárquica, em que primeiro é determinado o empacotamento das peças e em seguida o caminho de corte para ele, considerando o corte por peças através dos pontos de referência.

Na Tabela 3 são apresentados os resultados obtidos através da estratégia hierárquica e do MIEPICCP-VF. A primeira coluna contém o nome das instâncias. As colunas 2 - 8 apresentam os resultados obtidos pela estratégia hierárquica: o comprimento do empacotamento (Layout), o GAP (GAP_L = 100×(Melhor comprimento - Limitante inferior do comprimento)/Melhor comprimento) e o tempo necessário para obter o empacotamento (em segundos), o comprimento do caminho de corte (CC), o GAP (GAP_{CP} = 100×(Melhor CC - Limitante inferior do CC)/Melhor CC), o tempo utilizado para obter o caminho de corte (em segundos) e, finalmente, o tempo total (tempo para encontrar o empacotamento + tempo para encontrar o caminho de corte), respectivamente. As colunas 9 - 12 apresentam os resultados obtidos pelo MIEPICCP-VF: o comprimento do empacotamento (Layout), o comprimento do caminho de corte (CC), o GAP (GAP = 100×(Melhor função objetivo - Limitante inferior)/Melhor função objetivo) e o tempo para

Tabela 2 – Informações sobre as instâncias.

Instâncias	Placa		Peças	
	Altura	Comprimento	Tipos	Total
three	7	7	3	3
threep2	7	11	3	6
threep2w9	9	9	3	6
threep3	7	16	3	9
threep3w9	9	13	3	9
shapes4	13	24	4	4
shapes4N	20	24	4	4
shapes8	20	28	4	8
shapes2	40	16	4	8
blasz2	15	27	4	20
blazewicz1	15	8	7	7
blazewicz2	15	16	7	14
blazewicz3	15	22	7	21
rco1	15	8	7	7
rco2	15	17	7	14
rco3	15	25	7	21
fu5	38	18	4	5
fu6	38	24	5	6
fu7	38	24	6	7
fu8	38	24	7	8
J1-10-10-0	10	20	9	10
J1-10-10-1	10	19	10	10
J1-10-10-2	10	21	9	10
J1-10-10-3	10	23	10	10
J1-10-10-4	10	15	10	10
J1-12-20-0	20	12	11	12
J1-12-20-1	20	11	11	12
J1-12-20-2	20	14	12	12
J1-12-20-3	20	10	12	12
J1-12-20-4	20	16	12	12
J1-14-20-0	20	14	13	14
J1-14-20-1	20	14	13	14
J1-14-20-2	20	16	14	14
J1-14-20-3	20	12	13	14
J1-14-20-4	20	16	13	14

obter a solução, respectivamente. Vale a pena ressaltar que a função objetivo é uma combinação entre o empacotamento e o caminho de corte, conforme (5.20). Na última coluna, reportamos a melhoria apresentada no comprimento do caminho de corte para cada uma das instâncias em que o empacotamento de menor comprimento foi obtido (100×(CC da solução hierárquica - CC da solução integrada)/CC da solução hierárquica).

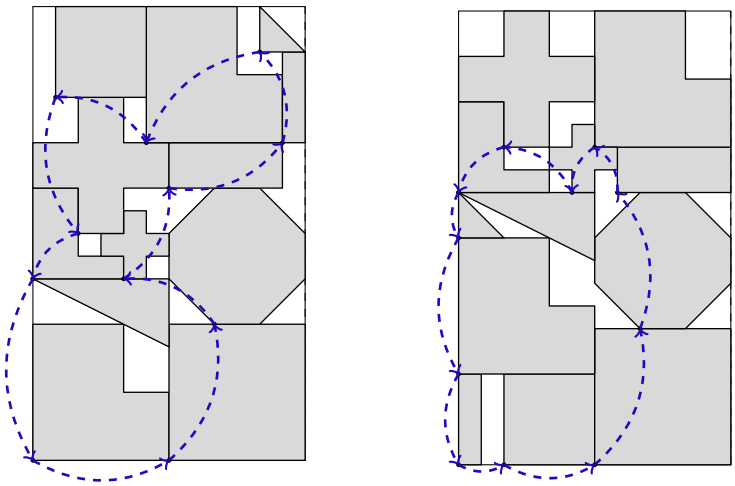
Analisando a Tabela 3, observamos que a estratégia hierárquica foi capaz de encontrar soluções factíveis para todas as instâncias. O empacotamento de menor comprimento foi encontrado para 33 instâncias, sendo que 29 também apresentaram o caminho de corte mínimo. O MIEPICCP-VF foi capaz de encontrar soluções factíveis para 33 instâncias, 24 com empacotamento de comprimento mínimo e 16 também apresentaram caminho de corte mínimo.

Tabela 3 – Resultados da estratégia hierárquica e do MIEPICCP-VF.

Instâncias	Estratégia hierárquica							MIEPICCP-VF				Melhoria
	Layout	**GAP_L (%)**	**Tempo (s)**	**CC**	**GAP_{CP} (%)**	**Tempo (s)**	**Tempo total (s)**	**Layout**	**CC**	**GAP (%)**	**Tempo (s)**	**CC (%)**
three	**6**	0,0	<1	17,1	0,0	<1	<1	**6**	**15,1**	0,0	<1	11,7
threep2	**10**	0,0	<1	22,8	0,0	<1	<1	**10**	**21,5**	0,0	2	5,7
threep2w9	**8**	0,0	<1	20,6	0,0	<1	<1	**8**	**20,6**	0,0	3	0,0
threep3	**14**	0,0	<1	36,0	0,0	2	2	**14**	**31,4**	0,0	13	12,9
threep3w9	**12**	0,0	<1	33,7	0,0	2	2	**12**	**26,0**	0,0	106	22,8
shapes4	**24**	0,0	1	33,7	0,0	<1	1	**24**	**31,5**	0,0	13	6,6
shapes4N	**14**	0,0	3	50,2	0,0	1	3	**14**	**48,1**	0,0	244	4,2
shapes8	**26**	0,0	64	74,9	0,0	3	67	**26**	**70,7**	0,0	1870	5,5
shapes2	**14**	0,0	3	86,9	0,0	7	10	**14**	**85,0**	0,4	TL	2,2
blasz2	**26**	0,0	23	92,5	0,0	8	31	27	82,4	30,2	TL	
blazewicz1	**8**	0,0	1	34,0	0,0	<1	1	**8**	**27,1**	0,0	836	20,5
blazewicz2	**14**	0,0	16	58,0	0,0	167	184	**14**	**47,3**	21,6	TL	18,4
blazewicz3	**20**	0,0	547	75,5	0,0	6	553	22	86,8	27,3	TL	
rco1	**8**	0,0	<1	31,6	0,0	<1	<1	**8**	**26,2**	0,0	552	16,9
rco2	**15**	0,0	3	57,1	0,0	2	5	**15**	**47,6**	15,2	TL	16,7
rco3	**22**	0,0	95	77,1	8,4	TL	TL	24	77,4	21,8	TL	
fu5	**18**	0,0	2	67,2	0,0	92	94	**18**	**57,1**	0,1	TL	15,1
fu6	**23**	0,0	328	76,4	0,0	1017	1345	**23**	**63,6**	0,1	TL	16,7
fu7	**24**	0,0	1832	75,9	0,0	569	2401	**24**	77,4	22,9	TL	-2,0
fu8	**24**	0,0	1242	98,8	0,0	1945	3187	-	-	-	TL	
J1-10-10-0	**18**	0,0	4	45,3	0,0	112	116	**18**	**45,0**	0,0	209	0,6
J1-10-10-1	**17**	0,0	10	37,9	0,0	59	69	**17**	**33,9**	0,0	968	10,8
J1-10-10-2	**20**	0,0	4	42,3	0,0	270	273	**20**	**35,7**	0,0	1081	15,5
J1-10-10-3	**21**	0,0	17	47,4	0,0	62	80	**21**	**40,4**	0,0	1723	14,9
J1-10-10-4	**13**	0,0	6	37,3	0,0	3	9	**13**	**27,7**	0,0	2415	25,8
J1-12-20-0	**12**	0,0	51	58,6	15,4	TL	TL	**12**	**41,3**	0,2	TL	29,5
J1-12-20-1	**10**	0,0	18	57,5	0,0	87	105	**10**	**41,1**	0,2	TL	28,5
J1-12-20-2	**12**	0,0	63	57,7	0,0	142	205	13	44,6	18,5	TL	
J1-12-20-3	**8**	0,0	22	45,3	5,4	TL	TL	**8**	**40,1**	0,0	2329	11,5
J1-12-20-4	**13**	0,0	427	60,2	0,0	175	602	14	51,2	20,8	TL	
J1-14-20-0	**12**	0,0	160	54,9	12,1	TL	TL	13	50,5	18,2	TL	
J1-14-20-1	**12**	15,5	TL	59,9	0,0	79	TL	-	-	-	TL	
J1-14-20-2	**14**	13,6	TL	68,1	0,0	421	TL	15	64,0	19,9	TL	
J1-14-20-3	**10**	0,0	26	54,7	0,0	149	175	11	45,0	21,0	TL	
J1-14-20-4	**14**	0,0	2111	59,8	0,0	472	2582	16	54,4	25,4	TL	
Média		0,8	408		1,2	579	987			8,0	2308	11,6

Comparando a estratégia hierárquica e o MIEPICCP-VF, temos que ambos encontraram empacotamentos de comprimento mínimo para 24 instâncias, contudo o MIEPICCP-VF foi capaz de encontrar empacotamentos que resultaram em caminhos de corte menores para 23 instâncias, enquanto 1 apresentou o mesmo comprimento de caminho de corte. No melhor caso, a melhoria no comprimento do caminho de corte foi de quase 30% (J1-12-20-0). O comprimento do caminho de corte da instância J1-12-20-0 foi $58,6$u.m. utilizando a estratégia hierárquica, enquanto que ele diminuiu para $41,3$u.m. utilizando o MIEPICCP-VF. Vale ressaltar que o comprimento do empacotamento é o mesmo, contudo a distribuição das peças no empacotamento é diferente, como ilustrado na Figura 27. Na Figura 27a é apresentado um empacotamento e o caminho de corte obtidos com a abordagem hierárquica, enquanto que na Figura 27b apresentado um empacotamento e o caminho de corte obtidos com o MIEPICCP-VF. Esses resultados demonstram que a integração é vantajosa mesmo quando impomos que o comprimento do empacotamento deve ser mínimo. Nesse caso, o modelo integrado encontra empacotamentos de comprimento mínimo que resultam também em caminhos de corte menores. Portanto, é reduzida a multiplicidade de soluções.

Figura 27 – Empacotamento e caminho de corte para a instância J1-12-20-0.



(a) Estratégia hierárquica. (b) MIEPICCP-VF.

Fonte: Elaborada pela autora.

Para as instâncias em que MIEPICCP-VF não foi capaz de encontrar empacotamento de comprimento mínimo (blaz2, blazewicz3, rco3, J1-12-20-2, J1-12-20-4, J1-14-20-0, J1-14-20-2, J1-14-20-3 e J1-14-20-4), o comprimento foi, em média, apenas $1,3$u.m. pior que aqueles encontrados pela estratégia hierárquica. Além disso, essas instâncias apresentaram GAPs altos de, em média, $22,6\%$. Analisando os tempos de execução, como já esperado, resolver hierarquicamente é mais fácil do que resolver os problema de maneira integrada. A estratégia hierárquica utilizou, em média, 987s para encontrar o empacotamento e o caminho de corte enquanto que MIEPICCP-VF demandou 2308s em média.

5.4.2 Fase II - Abordagem biobjetivo

Na segunda fase dos experimentos, avaliamos o MIEPICCP-VF como um modelo biobjetivo, analisando a influência que o empacotamento tem no comprimento do caminho de corte. Em modelos biobjetivos, não existe uma única solução ótima, mas um conjunto de soluções eficientes, que formam a chamada fronteira de Pareto. Utilizamos o método clássico ε-restrito para encontrar esse conjunto de soluções eficientes, sendo que o caminho de corte (5.5) permanece na função objetivo e o comprimento do empacotamento (5.4) é adicionado como restrição ao modelo ($z \leq \varepsilon$), i.e., as restrições (5.6) são substituídas por:

$$(c_d g_x + x_t^M)\delta_t^d \leq \varepsilon, \qquad \forall d \in \mathbb{IFP}_t, \forall t \in \mathbb{T}. \tag{5.21}$$

Para obter o conjunto de soluções eficientes, vários MIEPICCP-VF mono-objetivos precisam ser resolvidos, resultando em um processo custoso pois, como mostrado na Tabela 3, resolver o MIEPICCP-VF como um modelo mono-objetivo pode ser difícil e demorado.

Por esse motivo, limitamos esse experimento a apenas 4 instâncias pequenas utilizadas na primeira fase. As instâncias escolhidas são descritas na Tabela 4. O limitante superior para o comprimento da placa também foi aumentado para permitir maior flexibilidade nas soluções. Além disso, um computador mais potente, Intel Xeon E5-2620 (2.00GHz x 24) com 64GB de memória RAM e sistema operacional Ubuntu 12.04, foi utilizado e o tempo limite de 5 hora (TL = 18000s) foi considerado. Comparamos as soluções do modelo integrado com aquelas obtidas pela estratégia hierárquica.

Tabela 4 – Características do conjunto de instâncias.

Instâncias	**Placa**		**Peças**	
	Altura	**Comprimento**	**Tipos**	**Total**
three	7	9	3	3
threep2	7	15	3	6
threep3	7	20	3	9
shapes4	13	35	4	4
shapes4N	20	35	4	4

Na Tabela 5 são resumidos os resultados obtidos com a estratégia hierárquica e o MIEPICCP-VF biobjetivo. A primeira coluna apresenta o nome das instâncias. A segunda e a terceira colunas apresentam o comprimento dos empacotamentos e dos caminhos de corte obtidos com a estratégia hierárquica, respectivamente. Nas últimas cinco colunas, relatamos o valor ε utilizado, os comprimentos dos empacotamentos, dos caminhos de corte, o GAP e o tempo necessário para resolver o MIEPICCP-VF biobjetivo. Devido à discretização da placa, o parâmetro ε assume apenas valores inteiros pertencentes ao intervalo $[z_{min}, z_{max}]$. O valor z_{min} é igual ao menor comprimento de empacotamento, obtido pela estratégia hierárquica, enquanto que o valor z_{max} é obtido resolvendo primeiramente o MIEPICCP-VF considerando a função

objetivo (5.5), que retorna o menor caminho de corte ($\bar{dist}$), e depois resolvendo o MIEPICCP-VF considerando a função objetivo (5.4) e a restrição adicional

$$\sum_{i \in \mathbb{D}} \sum_{j \in \mathbb{D}} d_{ij} x_{ij} \leq \bar{dist}, \tag{5.22}$$

que retorna o menor comprimento z para o menor caminho de corte ($\bar{dist}$).

Tabela 5 – Resultados do MIEPICCP-VF biobjetivo comparado à estratégia hierárquica.

Instâncias	Estratégia hierárquica		MIEPICCP-VF				
	Layout	CC	ε	Layout	CC	GAP (%)	Tempo (s)
three	6	17,1	6	6	15,1	0,0	<1
			7	7	9,8	0,0	2
threep2	10	22,8	10	10	21,5	0,0	5
			11	11	18,8	0,0	85
threep3	14	36,0	14	14	31,4	0,0	25
			15	15	29,5	0,0	754
			16	16	29,0	13,2	TL
shapes4	24	33,7	24	24	31,5	0,0	65
shapes4N	14	50,2	14	14	48,1	0,0	40
			15	15	47,1	0,0	206
			16	16	41,2	0,0	461
			17	17	39,2	0,0	1573
			18	18	37,8	0,0	5130
			19	19	36,5	0,0	13723
			20	20	36,1	12,9	TL
			21	21	34,4	32,0	TL
			22	22	33,3	35,3	TL
			23	23	32,1	42,0	TL
			24	24	31,5	42,3	TL

Analisando a Tabela 5, notamos que quando o comprimento do empacotamento aumenta, a solução ótima para o comprimento do caminho de corte diminui. Por exemplo, o caminho de corte da instância three possui o comprimento de $15,1$u.m. quando o tamanho do empacotamento é limitado a 6u.m. e diminuiu para $9,8$u.m. quando o tamanho do empacotamento pode chegar até 7u.m. Em ambos os casos, os comprimentos dos caminhos de corte foram menores quando comparados à estratégia hierárquica, que foi de $17,1$u.m. para um empacotamento de comprimento igual a 6u.m. Observe também que, mesmo para instâncias com GAPs altos, os comprimentos dos caminhos de corte são menores e uma aproximação da fronteira de Pareto pode ser escrita para todas as instâncias.

É importante ressaltar que um empacotamento com comprimento de caminho de corte menor pode melhorar a produtividade de um dia de trabalho, pois o tempo economizado durante o processo de corte pode ser usado para cortar mais layouts. Por outro lado, isso pode levar a mais desperdício de material. Com essas soluções, o tomador de decisões encontrará a solução mais adequada para a empresa.

Por fim, também é possível que algumas instâncias não sejam biobjetivo, como é o caso da shapes4, que apresenta uma solução única que minimiza tanto o comprimento do empacotamento quanto o caminho de corte. Isso pode acontecer devido às características da instância. Por exemplo, no caso de shapes4, a altura da placa não permite o posicionamento das peças em empacotamentos alternativos que tenham comprimentos de caminho de corte menores. Portanto, propusemos a instância shapes4N que é igual a shapes4, exceto pela altura da placa, que foi aumentada de 13 para 20. Com essa alteração, a instância se torna mais flexível.

Novamente, como ambos os problemas são de difícil resolução, o problema integrado é pelo menos tão difícil quanto os problemas isolados. Isso pode ser observado pelo aumento no tempo necessário para resolver o modelo a medida que as instâncias aumentam de tamanho.

CAPÍTULO

6

MODELO INTEGRADO DE EMPACOTAMENTO DE PEÇAS IRREGULARES E CAMINHO DE CORTE POR PEÇA: VÉRTICE LIVRE

Neste capítulo, apresentamos a extensão do modelo integrado descrito na Seção 5.3, considerando o corte por peça a partir de um vértice qualquer, incluindo assim um maior número de possibilidades para inicio do corte[1]. Detalhes sobre esse modelo podem ser encontrados na Seção 6.1 e os experimentos computacionais são apresentados na Seção 6.2.

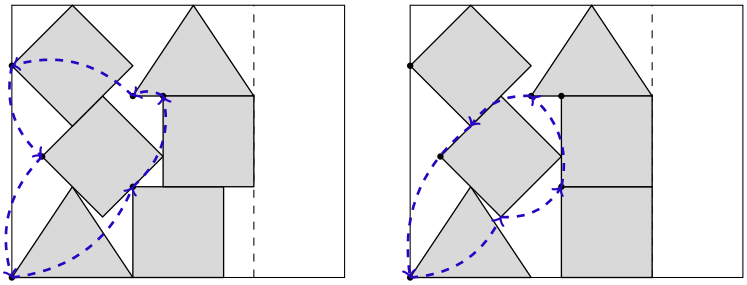
6.1 Modelo matemático estendido

Nesta seção, é apresentada uma extensão do modelo integrado de empacotamento de peças irregulares e caminho de corte por peça a partir de um vértice fixo, em que o corte ainda é feito por peça, porém não é necessário que o ponto de entrada e saída do corte seja o ponto de referência da peça, podendo ser agora qualquer um de seus vértices. A Figura 28 ilustra essa diferença, em que na Figura 28a o corte é realizado através de um vértice fixo pré-determinado (o ponto de referência da peça), enquanto na Figura 28b todos os vértices são opções para início do corte, porém apenas um é escolhido. Note que o caminho de corte continua definido pelo percurso feito pela ferramenta de corte entre os pontos de entrada de corte de todas as peças.

Desta forma, é preciso definir um conjunto de pontos pelos quais cada peça pode ter seu corte iniciado. Vale destacar que estes conjuntos dependem da localização em que o ponto de referência da peça está. Tem-se, portanto, o conjunto de pontos em que estão localizados todos os vértices da peça t quando o seu ponto de referência está localizado em d, denominado por $\mathbb{V}_t^d$.

[1] O texto desse capítulo é fortemente baseado na tradução de um artigo que está submetido a um periódico da área.

Figura 28 – Um exemplo de corte por vértice fixo e por vértice livre.

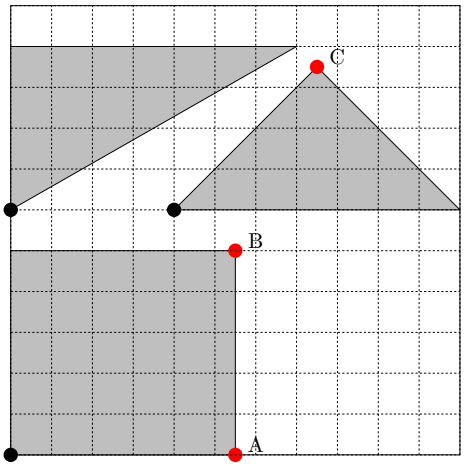


(a) Corte por vértice fixo. (b) Corte por qualquer vértice.

Fonte: Elaborada pela autora.

Note que os conjuntos $\mathbb{V}_t^d$ e $\mathbb{D}$ podem conter pontos extras além dos pontos da discretização da placa. Isso ocorre para os vértices das peças que não estão localizados nos pontos já existentes da malha, como ilustrado pelos vértices A, B e C na Figura 29. Nesses casos, novos pontos devem ser considerados, porém apenas nas restrições relacionadas à determinação do caminho de corte, não alterando em nada a discretização da placa utilizada no empacotamento das peças.

Figura 29 – Exemplo de vértices localizados fora dos pontos da malha regular.



Fonte: Elaborada pela autora.

Além dos índices, parâmetros e variáveis da Seção 5.3, considere também os conjuntos $\mathbb{V}_t^d$ e as variáveis:

- $y_{t,d}^i$: variável binária que recebe o valor 1 quando uma peça do tipo t com ponto de referência localizado no ponto d da placa possui ponto de entrada de corte no vértice localizado no ponto i e 0 caso contrário.

Para o modelo integrado de empacotamento de peças irregulares e caminho de corte por peça através de um vértice livre (MIEPICCP-VL), considere o modelo (5.4) – (5.19) e substitua as restrições (5.9) pelas restrições:

$$\sum_{i \in \mathbb{V}_t^d} y_{t,d}^i = \delta_t^d, \qquad \forall t \in \mathbb{T}, \forall d \in \mathbb{IFP}_t, \tag{6.1}$$

$$\sum_{t \in \mathbb{T}} \sum_{d \in \mathbb{IFP}_t} y_{t,d}^i \leq T_p w_i, \qquad \forall i \in \mathbb{D}. \tag{6.2}$$

As restrições (6.1) garantem que quando uma peça do tipo t tem seu ponto de referência alocado ao ponto d da placa, então exatamente um de seus vértices ($i \in \mathbb{V}_t^d$) é o ponto de entrada e saída do corte. As restrições (6.2) exigem que quando uma ou mais peças possuem seus pontos de entrada de corte localizados no ponto i, então a variável w_i assuma o valor 1, indicando que o caminho de corte deve incluir esse ponto.

O modelo MIEPICCP-VL é dado por:

$$\min \quad z \tag{6.3}$$

$$\min \quad \sum_{i \in \mathbb{D}} \sum_{j \in \mathbb{D}} d_{ij} x_{ij} \tag{6.4}$$

s.a

Restrições de empacotamento

$$(c_d g_x + X_t^M)\delta_t^d \leq z, \qquad \forall d \in \mathbb{IFP}_t, \forall t \in \mathbb{T}, \tag{6.5}$$

$$\sum_{d \in \mathbb{IFP}_t} \delta_t^d = q_t, \qquad \forall t \in \mathbb{T}, \tag{6.6}$$

$$\delta_u^e + \delta_t^d \leq 1, \qquad \forall e \in \mathbb{NFP}_{t,u}^d, \forall t, u \in \mathbb{T}, u \geq t, \forall d \in \mathbb{IFP}_t, \tag{6.7}$$

Restrições de acoplamento

$$\sum_{i \in \mathbb{V}_t^d} y_{t,d}^i = \delta_t^d, \qquad \forall t \in \mathbb{T}, \forall d \in \mathbb{IFP}_t, \tag{6.8}$$

$$\sum_{t \in \mathbb{T}} \sum_{d \in \mathbb{IFP}_t} y_{t,d}^i \leq T_p w_i, \qquad \forall i \in \mathbb{D}, \tag{6.9}$$

$$\sum_{i \in \mathbb{D}'} w_i \leq T_p, \tag{6.10}$$

Restrições de determinação de caminho de corte

$$\sum_{i \in \mathbb{D}, i \neq j} x_{ij} = w_j, \qquad \forall j \in \mathbb{D}', \tag{6.11}$$

$$\sum_{j \in \mathbb{D}, j \neq i} x_{ij} = w_i, \qquad \forall i \in \mathbb{D}', \tag{6.12}$$

$$\sum_{i \in \mathbb{D}'} x_{i0} = 1, \tag{6.13}$$

$$\sum_{j \in \mathbb{D}'} x_{0j} = 1, \tag{6.14}$$

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1, \qquad S \subseteq \{1, \ldots, D-1\}, \quad 1 \leq |S| \leq D-2, \tag{6.15}$$

Domínio das variáveis

$$\delta_t^d \in \{0,1\}, \qquad \forall d \in \mathbb{IFP}_t, \forall t \in \mathbb{T}, \tag{6.16}$$

$$w_i \in \{0,1\}, \qquad \forall i \in \mathbb{D}, \tag{6.17}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i,j \in \mathbb{D}, \tag{6.18}$$

$$y_{t,d}^i \in \{0,1\}, \qquad \forall d \in \mathbb{IFP}_t, \forall t \in \mathbb{T}, \forall i \in \mathbb{D}, \tag{6.19}$$

$$z \geq 0. \tag{6.20}$$

6.2 Experimentos computacionais

Seguindo a mesma estrutura dos experimentos realizados para o MIEPICCP-VF, apresentados na Seção 5.4, os testes computacionais utilizando o MIEPICCP-VL também foram divididos em duas fases. Na primeira fase, apresentada na Subseção 6.2.1, investigamos a melhoria no caminho de corte quando utilizamos o MIEPICCP-VL para obter o empacotamento de comprimento mínimo. Na segunda fase, descrita na Subseção 6.2.2, avaliamos o MIEPICCP-VL como um modelo biobjetivo, que busca minimizar o comprimento do empacotamento das peças e do caminho de corte. Novamente, os resultados foram comparados com os resultados obtidos pela estratégia hierárquica, em que primeiramente o empacotamento é definido e, em seguida, o seu caminho de corte é calculado.

6.2.1 Fase I - Empacotamento de comprimento mínimo

O objetivo dessa primeira fase é avaliar se o MIEPICCP-VL é capaz de encontrar empacotamentos de mesmo comprimento dos encontrados pelo modelo dos pontos original, mas com um comprimento menor para o caminho de corte. Para isso, consideramos a função objetivo (5.20).

Os testes computacionais foram realizados em um computador com processador Intel Core i7 7700 (3.60GHz x 8), 16GB de memória RAM e sistema operacional Ubuntu 16.04. O modelo matemático foi implementado em C/C++, usando o software de otimização ILOG CPLEX 12.6 tendo 1 hora (TL = 3600s) como tempo limite para resolução. Novamente, utilizamos as 35 instâncias da literatura dos problemas de corte e empacotamento apresentadas na Tabela 2. Primeiramente, os resultados obtidos ao resolver o MIEPICCP-VL foram comparados aos resultados da estratégia hierárquica, em que o empacotamento é definido primeiro e depois o caminho de corte é calculado. Em seguida, comparamos a estratégia de vértice livre *versus* a estratégia de vértice fixo, apresentada no Capítulo 5.

Na Tabela 6, são resumidos os resultados obtidos para a estratégia hierárquica e para o MIEPICCP-VL. Seguindo a mesma estrutura da Tabela 3, a primeira coluna contém o nome das instâncias, as colunas 2 - 8 trazem os resultados obtidos utilizando a estratégia hierárquica, com

Tabela 6 – Resultados da estratégia hierárquica e do MIEPICCP-VL.

Instâncias	Estratégia hierárquica							MIEPICCP-VL				Melhoria
	Layout	GAP_L (%)	Tempo (s)	CC	GAP_{CP} (%)	Tempo (s)	Tempo total (s)	Layout	CC	GAP (%)	Tempo (s)	CC (%)
three	**6**	0,0	<1	11,2	0,0	<1	<1	**6**	**9,8**	0,0	1	11,8
threep2	**10**	0,0	<1	16,8	0,0	37	37	**10**	**16,6**	0,0	3388	1,1
threep2w9	**8**	0,0	<1	16,8	0,0	20	20	**8**	**16,6**	0,0	221	1,1
threep3	**14**	0,0	<1	26,7	47,4	3180	3180	**14**	**23,4**	0,1	TL	12,4
threep3w9	**12**	0,0	<1	24,8	41,8	TL	TL	**12**	**23,4**	0,1	TL	5,8
shapes4	**24**	0,0	1	29,1	0,0	58	59	**24**	**27,3**	0,1	TL	6,0
shapes4N	**14**	0,0	3	33,7	0,0	626	628	**14**	**28,5**	0,0	3475	15,5
shapes8	**26**	0,0	64	64,9	67,7	TL	TL	28	75,4	42,6	TL	
shapes2	**14**	0,0	3	71,7	49,4	TL	TL	15	93,4	7,2	TL	
blasz2	**26**	0,0	23	90,1	75,5	TL	TL	-	-	-	TL	
blazewicz1	**8**	0,0	1	25,6	0,0	1234	1235	**8**	**21,6**	0,2	TL	15,5
blazewicz2	**14**	0,0	16	37,7	53,8	TL	TL	16	49,1	33,3	TL	
blazewicz3	**20**	0,0	547	75,2	72,9	TL	TL	-	-	-	TL	
rco1	**8**	0,0	<1	29,3	1,7	TL	TL	**8**	**21,1**	0,2	TL	27,9
rco2	**15**	0,0	3	40,9	60,1	TL	TL	17	49,7	26,4	TL	
rco3	**22**	0,0	95	79,2	72,9	TL	TL	25	98,6	25,0	TL	
fu5	**18**	0,0	2	65,9	63,3	TL	TL	**18**	90,4	0,5	TL	-37,1
fu6	**23**	0,0	328	65,3	61,1	TL	TL	-	-	-	TL	
fu7	**24**	0,0	1832	143,7	95,2	TL	TL	-	-	-	TL	
fu8	**24**	0,0	1242	101,5	82,4	TL	TL	-	-	-	TL	
J1-10-10-0	**18**	0,0	4	43,7	63,0	TL	TL	19	39,1	17,9	TL	
J1-10-10-1	**17**	0,0	10	31,3	47,6	TL	TL	**17**	**29,2**	0,1	TL	6,7
J1-10-10-2	**20**	0,0	4	37,0	58,8	TL	TL	**20**	**35,2**	0,2	TL	4,9
J1-10-10-3	**21**	0,0	17	37,6	64,1	TL	TL	22	51,1	20,0	TL	
J1-10-10-4	**13**	0,0	6	31,3	35,6	TL	TL	**13**	33,6	22,3	TL	-7,3
J1-12-20-0	**12**	0,0	51	50,3	67,4	TL	TL	**12**	51,4	16,8	TL	-2,1
J1-12-20-1	**10**	0,0	18	50,2	68,2	TL	TL	11	51,4	20,2	TL	
J1-12-20-2	**12**	0,0	63	49,0	69,2	TL	TL	14	58,7	24,5	TL	
J1-12-20-3	**8**	0,0	22	41,9	60,9	TL	TL	10	43,0	30,1	TL	
J1-12-20-4	**13**	0,0	427	49,3	65,9	TL	TL	-	-	-	TL	
J1-14-20-0	**12**	0,0	160	48,8	68,4	TL	TL	-	-	-	TL	
J1-14-20-1	**12**	15,5	TL	51,5	67,6	TL	TL	-	-	-	TL	
J1-14-20-2	**14**	13,6	TL	65,1	76,8	TL	TL	-	-	-	TL	
J1-14-20-3	**10**	0,0	26	55,3	69,0	TL	TL	12	47,8	27,7	TL	
J1-14-20-4	**14**	0,0	2111	57,1	71,7	TL	TL	-	-	-	TL	
Média		0,8	408,0		51,4	3027	3435			12,6	3391	4,4

as colunas 2 a 4 iguais às apresentadas na Tabela 3. Nas Colunas 9 - 12, são apresentados os resultados obtidos ao resolver o MIEPICCP-VL e na última coluna reportamos a melhoria do caminho de corte (100×(CC da estratégia hierárquica - CC da solução integrada)/CC da estratégia hierárquica) para as instâncias em que o empacotamento de comprimento mínimo foi encontrado. Os símbolos "**" indicam que o teste foi encerrado antes do tempo limite devido à limitação de memória. Novamente, vale mencionar que a função objetivo do MIEPICCP-VL é uma combinação do comprimento mínimo do empacotamento e do caminho de corte, dado que (5.20), então o GAP é calculado tomando o valor da função objetivo em conta, e não apenas o comprimento do empacotamento ou do caminho de corte.

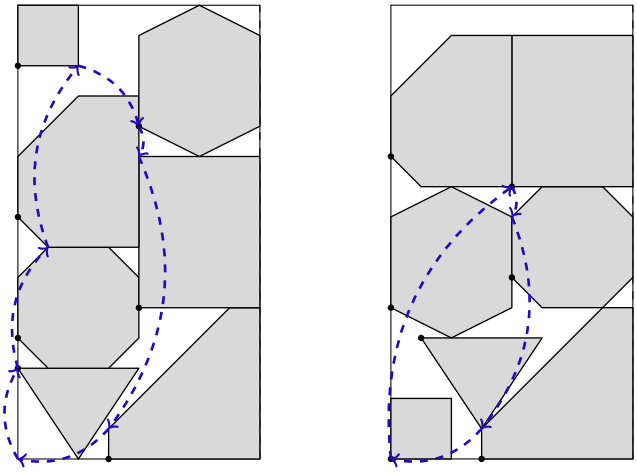
Analisando a Tabela 6, observamos que a estratégia hierárquica foi capaz de encontrar soluções factíveis para todas as instâncias. O comprimento mínimo do empacotamento foi encontrado em 33 instâncias (já discutidas na Tabela 3), sendo que para 6 delas também foi obtido o caminho mínimo de corte. O MIEPICCP-VL conseguiu encontrar soluções factíveis para 24 instâncias, das quais 14 resultaram em empacotamentos de mesmo comprimento obtido pela estratégia hierárquica, sendo 4 soluções ótimas.

Tanto a estratégia hierárquica quanto o MIEPICCP-VL encontraram o comprimento mínimo para o empacotamento em 14 instâncias, sendo que o MIEPICCP-VL foi capaz de encontrar empacotamentos com caminhos de corte menores para 11 instâncias e maiores em 3. No melhor caso, a melhoria no caminho foi próxima de 28% (rco1). O caminho de corte para a instância rco1 foi $29,2$u.m. com a estratégia hierárquica e diminuiu para $21,1$u.m. com o MIEPICCP-VL. Novamente, vale notar que o comprimento do empacotamento das peças é o mesmo, contudo o empacotamento é diferente, como ilustrado na Figura 30. Na Figura 30a é apresentado um empacotamento e o caminho de corte obtidos pela abordagem hierárquica, enquanto na Figura 30b é apresentado um empacotamento e caminho de corte obtidos com o MIEPICCP-VL.

Entre as 3 instâncias para as quais o MIEPICCP-VL encontrou caminhos de corte maiores (fu5, J1-10-10-4 e J1-12-20-0), a maior diferença aconteceu na fu5 (37%). Embora o GAP nessa instância seja de apenas $0,5\%$, há espaço para uma melhoria substancial no comprimento do caminho de corte, pois o GAP está relacionado com o valor da função objetivo ($18,0904 = 18 + 10^{-3} \times 90,4$), dado por (5.20), e não apenas pelo comprimento do empacotamento ou do caminho de corte.

Para as instâncias que o MIEPICCP-VL não conseguiu encontrar um empacotamento com comprimento mínimo (shapes8, shapes2, blazewicz2, rco2, rco3, J1-10-10-0, J1-10-10-3, J1-12-20-1, J1-12-20-2, J1-12-20-3 e J1-14-20-3), as soluções foram, em média, apenas 2u.m. piores do que as encontradas pela abordagem hierárquica, mas com altos GAPs de $25,7\%$, em média. Por fim, ambas as abordagens tiverem tempo computacional próximos, com médias de 3027s e 3391s para a estratégia hierárquica e o MIEPICCP-VL, respectivamente.

Figura 30 – Empacotamento e caminho de corte para a instância rco1.



(a) Estratégia hierárquica. (b) MIEPICCP-VL.

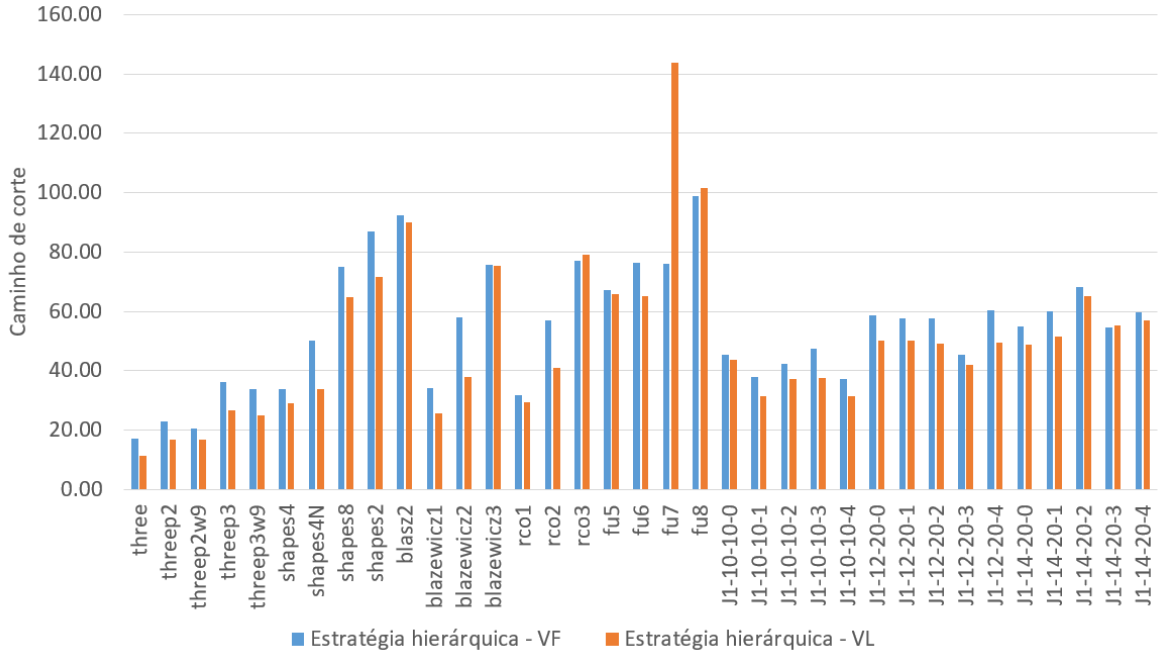
Fonte: Elaborada pela autora.

6.2.1.1 Comparação entre as estratégias de vértice fixo e de vértice livre

Primeiramente, analisaremos a influência da possibilidade de utilizar qualquer um dos vértices das peças como início do corte (versus) o vértice fixo, dentro da estratégia hierárquica. Comparando a Tabela 3 e a Tabela 6, temos que a estratégia hierárquica considerando o corte por qualquer vértice (vértice livre) não foi capaz de melhorar o comprimento do caminho de corte, dentro do tempo limite de 1 hora, para as instâncias rco3, fu7, fu8 e J1-14-20-3, quando comparada com a sua versão considerando o corte apenas pelo ponto de referência, como é ilustrado na Figura 31. Embora o corte por qualquer vértice possa diminuir o comprimento do caminho de corte, já que adiciona mais flexibilidade ao problema, ele também torna o modelo mais difícil de ser resolvido. O caminho de corte encontrado pela estratégia hierárquica considerando o corte por qualquer vértice teve, em média, um GAP de $51,4\%$ e tempo computacional de 3027s, com 28 instâncias atingindo o tempo limite, enquanto as soluções encontradas para sua versão simplificada tiveram, em média, um GAP de $1,2\%$ e tempo de 579s, e apenas 4 instâncias atingiram o tempo limite.

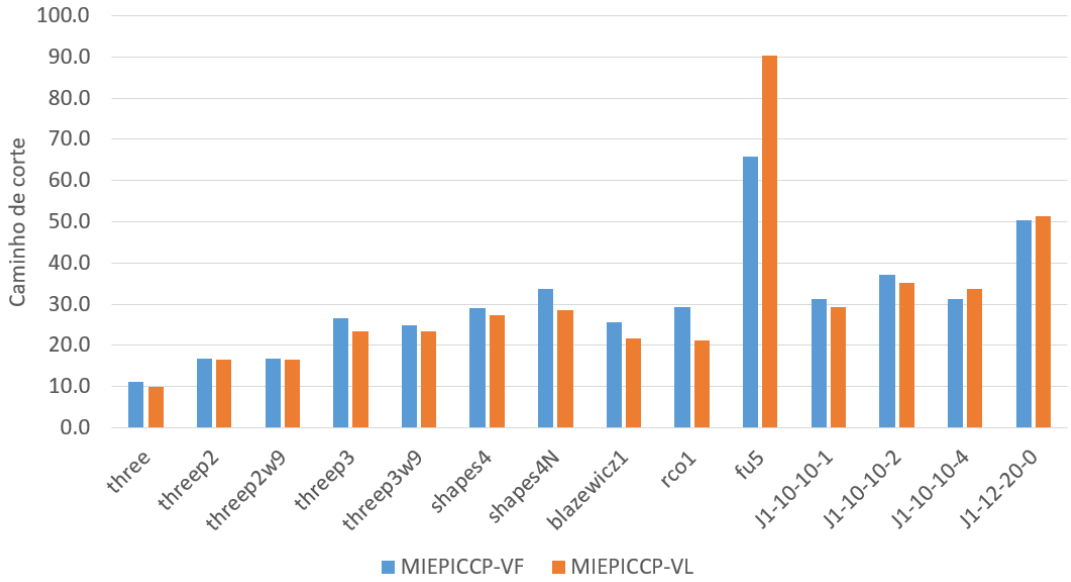
Para analisar as performances do MIEPICCP-VL e MIEPICCP-VF, consideramos apenas os caminhos de corte das instâncias para as quais o empacotamento com comprimento mínimo foi encontrado para ambos os modelos, pois, diferentemente da estratégia hierárquica, os empacotamentos não são os mesmos. Embora o MIEPICCP-VL tenha conseguido melhorar o comprimento de quase todos os caminhos de corte dentro do tempo limite de 1 hora (com exceção de fu5, J1-10-10-4 e J1-12-20-0), como ilustrado na Figura 32, a média do GAP e do tempo computacional ($12,6\%$ e 3391s, respectivamente) foram piores que os resultados obtidos pelo MIEPICCP-VF (8% e 2308s, respectivamente). O número de instâncias que chegou ao tempo limite também foi maior para o MIEPICCP-VL (31) do que para o MIEPICCP-VF (19).

Figura 31 – Comparação entre os comprimentos do caminho de corte obtidos pelas estratégias hierárquicas.



Fonte: Elaborada pela autora.

Figura 32 – Comparação entre os comprimentos do caminho de corte obtidos por MIEPICCP-VF e MIEPICCP-VL.



Fonte: Elaborada pela autora.

Conclusão, cortar a peça por qualquer um de seus vértices adiciona flexibilidade ao problema e aumenta o conjunto de soluções factíveis. Como esperado, no entanto, o modelo resultante é mais difícil de ser resolvido do que o modelo que considera o corte apenas através dos pontos de referência.

6.2.2 Fase II - A abordagem biobjetivo

Na segunda fase dos experimentos, avaliamos o MIEPICCP-VL como um modelo biobjetivo, analisando a influência que o empacotamento tem no comprimento do caminho de corte. Assim como na Subseção 5.4.2, utilizamos o método clássico ε-restrito para encontrar um conjunto de soluções eficientes, sendo que o caminho de corte (5.5) permanece na função objetivo e o comprimento do empacotamento (5.4) é adicionado como restrição ao modelo.

Mantemos as mesmas características utilizadas na Subseção 5.4.2: as 4 instâncias descritas na Tabela 4, o computador Intel Xeon E5-2620 (2.00GHz x 24) com 64GB de memória RAM e sistema operacional Ubuntu 12.04 e o tempo limite de 5 hora (TL = 18000s). Comparamos as soluções do modelo integrado com aquelas obtidas pela estratégia hierárquica. Na Tabela 7, apresentamos os resultados obtidos para o MIEPICCP-VL e a estratégia hierárquica, seguindo a mesma estrutura da Tabela 5.

Tabela 7 – O MIEPICCP-VL biobjetivo comparado à estratégia hierárquica.

Instâncias	**Estratégia hierárquica**		**MIEPICCP-VL**				
	Layout	**CC**	ε	**Layout**	**CC**	**GAP (%)**	**Tempo (s)**
three	6	11,2	6	6	9,8	0,0	2
			7	7	8,5	0,0	55
threep2	10	16,8	10	10	16,6	38,4	TL
			11	11	15,4	49,4	TL
threep3	14	26,7	14	14	23,4	47,9	TL
			15	15	25,5	71,7	TL
			16	16	25,9	78,1	TL
shapes4	24	29,1	24	24	27,3	57,0	TL
shapes4N	14	33,7	14	14	28,5	0,0	5843
			15	14	28,6	67,6	TL
			16	16	23,9	74,4	TL
			17	17	29,7	90,6	TL
			18	18	25,8	86,1	TL
			19	19	24,4	83,0	TL
			20	19	29,1	89,7	TL
			21	19	27,8	88,6	TL
			22	20	22,7	81,8	TL
			23	20	24,4	82,9	TL
			24	24	24,7	83,2	TL

Analisando a Tabela 7, notamos que, analogamente ao MIEPICCP-VF, quando o comprimento do empacotamento era maior, o comprimento do caminho de corte era reduzido. Por

exemplo, o caminho de corte da instância three foi de $9,8$u.m. para o empacotamento de comprimento igual a 6u.m., enquanto que o caminho diminuiu para $8,5$u.m. quando o comprimento do empacotamento aumentou para 7u.m.. Em ambos os casos, os comprimento dos caminhos de corte foram menores quando comparados à estratégia hierárquica, que foi de $11,2$u.m. para o empacotamento de 6u.m. A fronteira de Pareto foi encontrada para a instância three. Para todos os outros casos, foi muito difícil resolver o MIEPICCP-VL, principalmente devido ao aumento de possibilidades para o caminho de corte.

Visando facilitar a leitura, a Tabela 8 apresenta os resultados obtido pelo MIEPICCP-VF, apresentados na Tabela 5, e pelo MIEPICCP-VL, apresentados na Tabela 7, ambos utilizando a abordagem biobjetiva. Como esperado, todos os caminhos de corte do MIEPICCP-VL foram melhores quando comparados com os obtidos com o MIEPICCP-VF, indicando assim, que utilizar a estratégia de vértice livre é melhor do que utilizar a estratégia de vértice fixo. Contudo, como ambos os problemas são difíceis de resolver, o problema integrado é tão difícil quanto os problemas isolados e isso pode ser observado pelo aumento do tempo necessário para resolver os modelos à medida que as instâncias aumentam de tamanho. Assim, adicionar mais possibilidades ao caminho de corte acrescenta mais dificuldade ao modelo.

Tabela 8 – Comparação entre o MIEPICCP-VF e MIEPICCP-VL biobjetivos.

Instâncias		**MIEPICCP-VF**				**MIEPICCP-VL**			
	ε	**Layout**	**CC**	**GAP (%)**	**Tempo (s)**	**Layout**	**CC**	**GAP (%)**	**Tempo (s)**
three	6	6	15,1	0,0	<1	6	9,8	0,0	2
	7	7	9,8	0,0	2	7	8,5	0,0	55
threep2	10	10	21,5	0,0	5	10	16,6	38,4	TL
	11	11	18,8	0,0	85	11	15,4	49,4	TL
threep3	14	14	31,4	0,0	25	14	23,4	47,9	TL
	15	15	29,5	0,0	754	15	25,5	71,7	TL
	16	16	29,0	13,2	TL	16	25,9	78,1	TL
shapes4	24	24	31,5	0,0	65	24	27,3	57,0	TL
shapes4N	14	14	48,1	0,0	40	14	28,5	0,0	5843
	15	15	47,1	0,0	206	14	28,6	67,6	TL
	16	16	41,2	0,0	461	16	23,9	74,4	TL
	17	17	39,2	0,0	1573	17	29,7	90,6	TL
	18	18	37,8	0,0	5130	18	25,8	86,1	TL
	19	19	36,5	0,0	13723	19	24,4	83,0	TL
	20	20	36,1	12,9	TL	19	29,1	89,7	TL
	21	21	34,4	32,0	TL	19	27,9	88,6	TL
	22	22	33,3	35,3	TL	20	22,7	81,8	TL
	23	23	32,1	42,0	TL	20	24,4	82,9	TL
	24	24	31,5	42,3	TL	24	24,7	83,2	TL

Parte II

Abordagens com empacotamento contínuo e corte por arestas

CAPÍTULO

7

MATHEURÍSTICA PARA O EMPACOTAMENTO DE PEÇAS IRREGULARES EM FAIXAS

Neste capítulo, propomos uma matheurística evolutiva para o problema de empacotamento de peças irregulares em faixas em domínio contínuo. No empacotamento em domínio contínuo, o posicionamento das peças no objeto é livre, ou seja, não existem pontos pré-definidos para o posicionamento dessas peças. A matheurística é detalhada na Seção 7.1 e, por se tratar de uma matheurística evolutiva, vários parâmetros influenciam sua eficiência, assim utilizamos o pacote *irace* para obter uma boa configuração de seus parâmetros. Detalhes sobre este pacote são apresentados na Seção 7.2. Por fim, os resultados dos testes computacionais são discutidos na Seção 7.3.

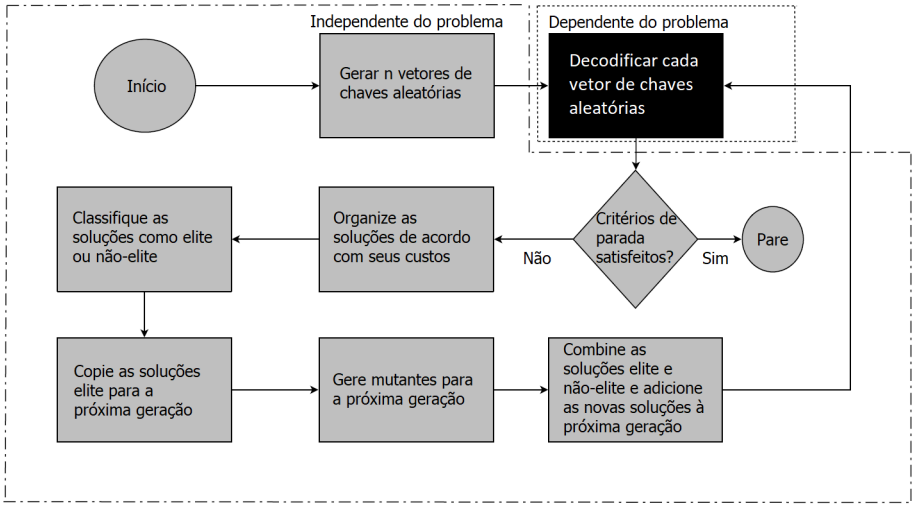
7.1 Uma matheurística para o empacotamento de peças irregulares em faixas em domínio contínuo

A metaheurística desenvolvida tomou como base a metaheurística evolutiva BRKGA (*biased random-key genetic algorithm*), proposta por Gonçalves e Resende (2011). No BRKGA, cada solução é representada por um vetor de n chaves aleatórias, em que cada chave é um número real aleatório dentro do intervalo contínuo [0; 1). Um decodificador é responsável por transformar cada vetor de chaves aleatórias em uma solução do problema de otimização e retornar um valor representativo para a mesma.

O BRKGA pode, a princípio, ser utilizado para resolver qualquer tipo de problema de otimização, pois é dividido em duas partes: uma independente e outra dependente do problema de otimização, como ilustrado na Figura 33. A parte dependente do problema é responsável por decodificar os vetores de chaves aleatórias e retornar um valor de avaliação como, por exemplo,

o custo associado a cada solução, enquanto a parte independente é responsável pela evolução e não requer nenhum conhecimento sobre o problema que está sendo resolvido.

Figura 33 – Framework do BRKGA.



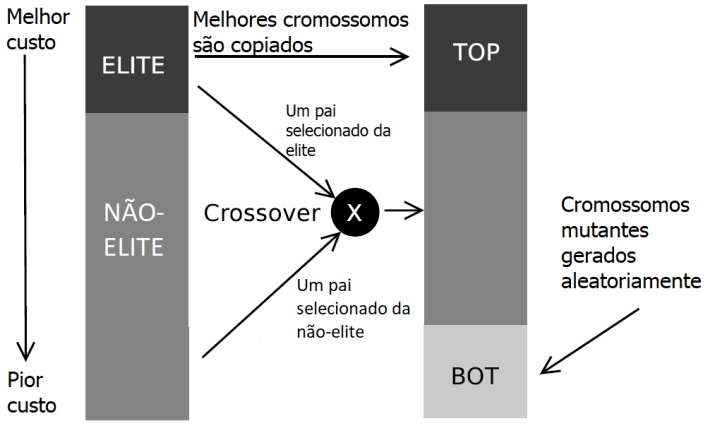
Fonte: Adaptada de Gonçalves e Resende (2011).

Na Figura 34 é ilustrada a dinâmica evolutiva do BRKGA, em que à esquerda temos a geração atual k e à direita a geração seguinte $k+1$. De maneira geral, o algoritmo do BRKGA é iniciado com uma população de p vetores de chaves aleatórias, também chamados de cromossomos. A cada geração, os cromossomos são particionados em dois conjuntos: um formado pelos pe ($pe < p - pe$) cromossomos com melhores custos, chamado de conjunto elite, e outro com o restante da população ($p - pe$), chamado de conjunto não-elite. Todos os elementos da elite são copiados para a população da próxima geração. Um número pequeno pm de vetores de chaves aleatórias, denominado mutantes, também é adicionado à população da próxima geração (BOT). Por fim, o restante da população ($p - pe - pm$) da geração $k+1$ é composta por cromossomos gerados por um procedimento de cruzamento (*crossover*) entre pares de cromossomos, em que um é sempre escolhido dentro do conjunto elite e o outro do conjunto não-elite. A seleção dos pais é feita de maneira aleatória e com reposição.

Na Figura 35, é apresentado um exemplo que ilustra o processo de *crossover* entre dois cromossomos com cinco genes cada. O primeiro vetor representa um cromossomo elite e o segundo um cromossomo não-elite. Nesse exemplo, consideramos a probabilidade de 70% do descendente herdar um alelo do cromossomo elite. Um número real, gerado aleatoriamente no intervalo [0; 1), simula o lançamento de uma moeda viciada. Se o resultado for menor que $0,7$, então o novo cromossomo herda o alelo do cromossomo elite, senão ele herda o alelo do cromossomo não-elite. No exemplo, o cromossomo descendente recebe alelos do cromossomo elite na primeira, quarta e quinta posições do vetor, e recebe alelos do cromossomo não-elite na segunda e terceira posições do vetor.

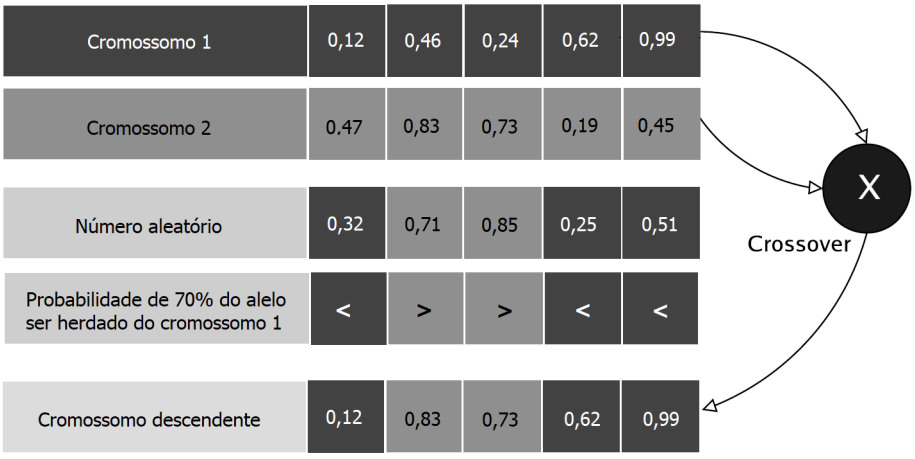
Com a população completa, o decodificador calcula o custo de todos os novos cromos-

Figura 34 – Dinâmica evolutiva do BRKGA.



Fonte: Adaptada de Gonçalves e Resende (2011).

Figura 35 – *Crossover* entre dois cromossomos.



Fonte: Elaborada pela autora.

somos adicionados, a população é dividida em novos conjuntos elite e não-elite e o algoritmo inicia uma nova geração. O algoritmo é interrompido quando um critério de parada for atingido e a solução do primeiro cromossomo elite é fornecida como a melhor solução encontrada.

Para adaptar o BRKGA a qualquer problema de otimização é necessário alterar apenas a parte do algoritmo que é dependente do problema, ou seja, i) definir o que é representado pelo cromossomo, e assim o seu tamanho; e ii) elaborar um decodificador.

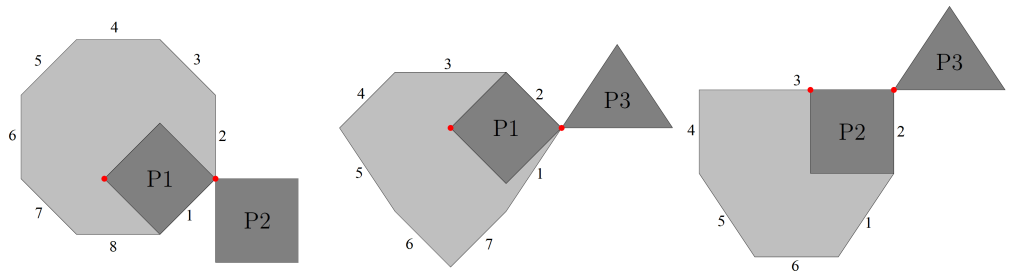
7.1.1 Cromossomo

Neste método, definimos que o tamanho do vetor de chaves aleatórias do BRKGA é dado pelo total de *no-fit polygons* entre as peças i e j. Dado que as peças não convexas são divididas em partes convexas, o tamanho do cromossomo é, portanto, igual a soma de todas as partes de todos os NFP entre as peças i e j. Assim, cada posição do vetor codifica qual aresta da parte p

do NFP_{ij}^{p} é utilizada para garantir a não sobreposição entre as peças i e j.

Na Figura 36, é apresentado um exemplo, em que na Figura 36a temos as peças $P1$, $P2$, $P3$, os *no-fit polygons* entre cada par de peças e suas arestas numeradas em sentido anti-horário. Como todas as peças podem ser representadas por apenas um polígono convexo então os *no-fit polygons* são compostos de apenas uma parte convexa e têm um total de 8, 7 e 6 arestas, respectivamente. Assim, como pode ser visto na Figura 36b, o cromossomo para esta instância tem tamanho 3 e cada chave aleatória é decodificada em uma aresta do *no-fit polygon*. Para o primeiro alelo, temos que o intervalo [0; 1) deve ser dividido igualmente entre as 8 arestas do NFP_{12}, ou seja, os números reais dentro do intervalo [0; 0,125) são decodificados na aresta 1 do NFP_{12}, números dentro do intervalo [0,125; 0,25) são decodificados na aresta 2 do NFP_{12} e assim sucessivamente. O raciocínio é análogo para os demais alelos do cromossomo.

Figura 36 – Cromossomo para matheurística proposta.



(a) Peças $P1$, $P2$, $P3$ e seus respectivos *no-fit polygons*.

Cromossomo	NFP_{12}	NFP_{13}	NFP_{23}
Chaves aleatórias	0,37	0,13	0,79
Arestas decodificas	3ª	1ª	5ª

(b) Estrutura, codificação e decodificação do cromossomo.

Fonte: Elaborada pela autora.

7.1.2 Decodificador

Para o decodificador da matheurística desenvolvida foi utilizado o modelo linear proposto por Cherri *et al.* (2016), que é um dos mais recentes para o empacotamento de peças irregulares em domínio contínuo. Neste modelo, as peças são decompostas em partes convexas e os *no-fit polygons* são gerados a partir delas. As principais vantagens de utilizar este modelo, segundo os autores, são a sua capacidade de encontrar soluções para instâncias cujas peças possam ser descritas por polígonos e o fato deste modelo usar ferramentas geométricas mais simples que os demais modelos existentes na literatura, tornando-o, assim, mais facilmente implementável e

livre de instabilidade numérica. Os índices, os parâmetros e as variáveis do modelo de Cherri *et al.* (2016) estão definidos a seguir.

Conjuntos, índices e parâmetros

- $N = \sum_{t=1}^{T} d_t$: total de peças a serem cortadas, em que T é o total de tipos de peças e d_t é a demanda de cada tipo, com $t \in \{1, \ldots, T\}$;
- $i, j \in \{1, \ldots, N\}$: índices de peças;
- NFP_{ij}: *no-fit polygon* entre as peças i e j;
- $\mathbb{Q}_{ij} = \{1, \ldots, Q_{ij}\}$: conjunto de partes que compõem o NFP_{ij};
- NFP_{ij}^{p}: parte p do *no-fit polygon* (NFP_{ij}), $p \in \mathbb{Q}_{ij}$;
- $\mathbb{K}_{ij}^{p} = \{1, \ldots, K_{ij}^{p}\}$: conjunto de arestas direcionadas do NFP_{ij}^{p};
- $k \in \mathbb{K}_{ij}^{p}$: índice para uma aresta direcionada do NFP_{ij}^{p};
- H: altura da placa;
- l_i^{min} e l_i^{max}: são as distâncias horizontais entre o ponto de referência da peça i e o vértice mais a esquerda e mais a direita da mesma peça, respectivamente;
- h_i^{min} e h_i^{max}: são as distâncias verticais entre o ponto de referência da peça i e o vértice mais acima e mais abaixo da mesma peça, respectivamente;
- $(c_{ij,x}^{p}, c_{ij,y}^{p})$ e $(d_{ij,x}^{p}, d_{ij,y}^{p})$: dois vértices consecutivos do NFP_{ij}^{p}.

Variáveis

- L: variável que representa o comprimento utilizado da placa;
- x_i: variável real que recebe o valor da coordenada x do ponto de referência da peça i;
- y_i: variável real que recebe o valor da coordenada y do ponto de referência da peça i;
- v_{ij}^{pk}: variável inteira que recebe 1 se o ponto de referência da peça j está a direita ou sobre a aresta k do NFP_{ij}^{p} e 0 caso contrário.

O modelo de otimização inteira mista para o empacotamento de peças irregulares em faixa com domínio contínuo de Cherri *et al.* (2016) é dado por:

$$\min \quad L \tag{7.1}$$

s.a:

$$l_i^{min} \leq x_i \leq L - l_i^{max}, \qquad i \in \{1,\ldots,N\}, \tag{7.2}$$

$$h_i^{min} \leq y_i \leq H - h_i^{max}, \qquad i \in \{1,\ldots,N\}, \tag{7.3}$$

$$\begin{aligned}&(c_{ij,x}^p - d_{ij,x}^p)c_{ij,y}^p - (c_{ij,y}^p - d_{ij,y}^p)c_{ij,x}^p - (c_{ij,x}^p - d_{ij,x}^p)(y_j - y_i)\\ &+ (c_{ij,y}^p - d_{ij,y}^p)(x_j - x_i) \leq (1 - v_{ij}^{pk})M_{ij}^{pk}, \qquad i \in \{1,\ldots,N-1\}, p \in \mathbb{Q}_{ij},\\ &\qquad j \in \{i+1,\ldots,N\}, k \in \mathbb{K}_{ij}^p,\end{aligned} \tag{7.4}$$

$$\sum_{k \in K_{ij}} v_{ij}^{pk} = 1, \qquad i \in \{1,\ldots,N-1\}, \quad j \in \{i+1,\ldots,N\}, p \in \mathbb{Q}_{ij}, \tag{7.5}$$

$$x_i, y_i \in \mathbb{R}_+, \qquad i,j \in \{1,\ldots,N\}, \tag{7.6}$$

$$v_{ij}^{pk} \in \{0,1\}, \qquad i,j \in \{1,\ldots,N\}, \quad p \in \mathbb{Q}_{ij}, k \in \mathbb{K}_{ij}^p. \tag{7.7}$$

O modelo (7.1) – (7.7) visa minimizar o comprimento total da placa utilizada na alocação das peças. As restrições (7.2) e (7.3) são responsáveis por garantir que todas as peças estejam inteiramente contidas dentro da placa. As restrições (7.4) garantem que não há sobreposição entre as peças e são escritas com base no *no-fit polygon* entre as peças i e j (NFP_{ij}). Para garantir a não sobreposição entre as peças, o ponto de referência da peça j deve estar na região externa do NFP_{ij}^p, para todo p, $p \in \mathbb{Q}_{ij}$. Essas restrições são satisfeitas quando impomos que o ponto de referência da peça j esteja à direita de exatamente uma aresta k, $k \in \mathbb{K}_{ij}^p$. As restrições (7.5) garantem que exatamente uma aresta k, $k \in \mathbb{K}_{ij}^p$ separa as peças i e j. Por fim, as restrições (7.6) e (7.7) definem o domínio das variáveis do modelo.

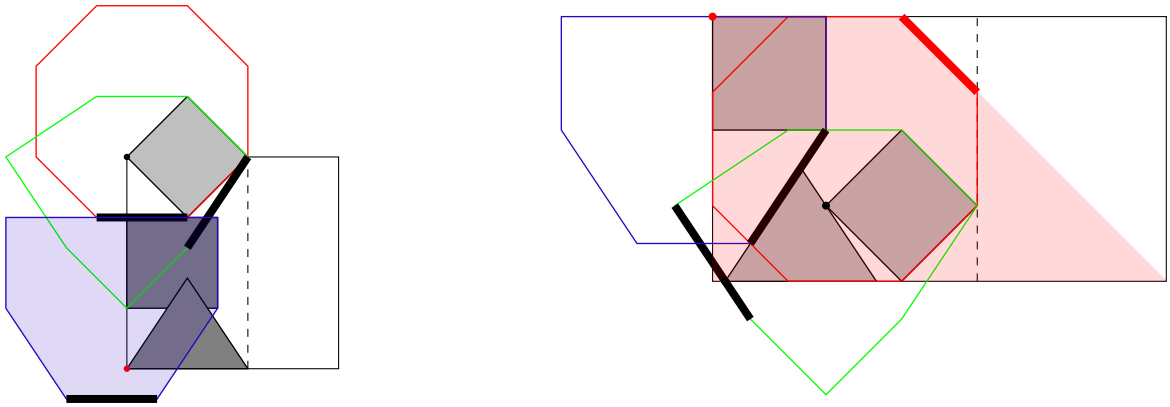
Note que existe uma variável inteira v_{ij}^{pk} para cada aresta de cada uma das partes convexas p de cada NFP_{ij}, além de uma restrição adicionada ao modelo para cada uma destas variáveis. Contudo, é necessário que apenas uma dessas restrições esteja ativa para cada par de peças, ou seja, para cada NFP_{ij}. Assim, o método aqui proposto visa reduzir o problema a um modelo linear contínuo deixando a cargo do BRKGA, através do cromossomo, decidir quais arestas devem ser utilizadas para garantir a não sobreposição entre as peças, ou seja, quais variáveis binárias têm valor igual a um. Com isso, o modelo (7.1) – (7.7) torna-se um modelo de programação linear, que é mais fácil e rápido de ser resolvido na otimalidade.

Contudo, algumas modificações se fazem necessárias. Primeiro, note que a decodificação do cromossomo pode gerar soluções infactíveis, principalmente nas gerações iniciais do BRKGA. Na Figura 37a, é apresentado um exemplo utilizando as mesmas peças e *no-fit polygons* apresentados na Figura 36. Neste exemplo, o cromossomo [0,964375; 0212519; 0,958365] foi decodificado nas arestas 8, 1 e 6, respectivamente, que encontram-se destacadas em negrito. Tal

decodificação gera uma solução com sobreposição entre as peças $P2$ (quadrado) e $P3$ (triângulo).

Outra situação que pode ocorrer ao longo das gerações é apresentada na Figura 37b. O cromossomo [0,373034; 0,660195; 0,11398] foi decodificado nas arestas 3, 5 e 1, respectivamente, que encontram-se destacadas. Note que não há sobreposição entre as peças porém, há infactibilidade nas restrições do modelo, uma vez que o ponto de referência da peça $P2$ (quadrado) encontra-se à direita da reta suporte da aresta do NFP_{12} (destacada em vermelho), que está sendo utilizada na restrição de não sobreposição.

Figura 37 – Exemplo de soluções obtidas pelo decodificador.



(a) Solução com sobreposição.

(b) Solução sem sobreposição, porém não viável.

Fonte: Elaborada pela autora.

Para permitir que estas soluções sejam viáveis, optamos por adicionar variáveis artificiais de folga (a_{ij}) às restrições de não sobreposição, que permitem que existam sobreposições entre as peças i e j. Tais variáveis são adicionadas à função objetivo, fazendo com que o foco do modelo seja encontrar uma solução factível para o problema original e de menor comprimento. De maneira análoga, optamos por adicionar uma variável artificial de folga (b_i) às restrições que garantem que as peças i estejam inteiramente contidas dentro da placa. A inclusão de tal variável visa que o método tenha mais liberdade para movimentar as peças e assim consiga melhores empacotamentos. Assim como a variável de folga das restrições de não sobreposição, esta também é adicionada à função objetivo do modelo. Desta forma, a função objetivo (7.1) é substituída por

$$\min \quad \sum_{i=1}^{N} b_i + \sum_{i=1}^{N} \sum_{j=i}^{N} a_{ij} + 10^{-6} L \tag{7.8}$$

que visa minimizar o comprimento total da placa utilizada na alocação das peças e a infactibilidade da solução gerada por cada cromossomo. As restrições (7.3) e (7.4) são substituídas por

$$h_i^{min} \leq y_i \leq H - h_i^{max} + b_i, \qquad i \in \{1,\ldots,N\}, \quad (7.9)$$

$$(c_{ij,x}^p - d_{ij,x}^p)c_{ij,y}^p - (c_{ij,y}^p - d_{ij,y}^p)c_{ij,x}^p - (c_{ij,x}^p - d_{ij,x}^p)(y_j - y_i)$$
$$+ (c_{ij,y}^p - d_{ij,y}^p)(x_j - x_i) - a_{ij} \leq 0, \qquad i \in \{1,\ldots,N-1\}, j \in \{i+1,\ldots,N\}, p \in \mathbb{Q}_{ij}, \quad (7.10)$$

respectivamente. Note que, como as variáveis inteiras estão fixadas, elas não são necessárias para o modelo.

7.1.3 Heurística de melhoria

Após o algoritmo ser encerrado tendo atingido o máximo de gerações permitidas, aplicamos uma heurística de busca local na melhor solução encontrada. Na heurística, a vizinhança da solução é definida permitindo mudar em parte a posição relativa das peças, ou seja, os valores de v_{ij}^{pk} são considerados variáveis.

7.1.3.1 Vizinhança da busca local

Dada a melhor solução encontrada, definimos os conjuntos A^+ e A^- a partir dos valores de v_{ij}^{pk}, em que $i \in \{1,\ldots,N-1\}, j \in \{i+1,\ldots,N\}, p \in \mathbb{Q}_{ij}$ e $k \in K_{ij}$, cujos valores são 0 e 1, respectivamente.

Uma vez particionadas as variáveis, a restrição (7.11) representa a distância da solução atual para uma nova solução. Esta distância é chamada de distância de Hamming e limita a quantidade de alelos que podem ser alterados na solução para obter uma solução vizinha da atual.

$$\sum_{a \in A^+} v_a + \sum_{a \in A^-} (1 - v_a) \leq 2\eta. \quad (7.11)$$

7.1.3.2 Estratégia de busca

Três estratégias de busca na vizinhança foram analisadas: i) descida; ii) máxima descida; e iii) *Proximity Search*. As três foram realizadas tendo como base o modelo (7.2) – (7.7) limitado a vizinhança escolhida, como descrito a seguir. Como o objetivo é encontrar uma solução vizinha de melhor qualidade, a restrição (7.12) é adicionada ao modelo.

$$L \leq \bar{L} - \theta. \quad (7.12)$$

Portanto, o modelo utilizado durante a busca local é dado por:

$$\min \quad L \tag{7.13}$$

s.a:

$$l_i^{min} \leq x_i \leq L - l_i^{max}, \qquad i \in \{1,\ldots,N\}, \tag{7.14}$$

$$h_i^{min} \leq y_i \leq H - h_i^{max}, \qquad i \in \{1,\ldots,N\}, \tag{7.15}$$

$$\begin{aligned}&(c_{ij,x}^p - d_{ij,x}^p)c_{ij,y}^p - (c_{ij,y}^p - d_{ij,y}^p)c_{ij,x}^p - (c_{ij,x}^p - d_{ij,x}^p)(y_j - y_i)\\ &+ (c_{ij,y}^p - d_{ij,y}^p)(x_j - x_i) \leq (1 - v_{ij}^{pk})M_{ij}^{pk}, \qquad i \in \{1,\ldots,N-1\}, p \in \mathbb{Q}_{ij},\\ &\qquad j \in \{i+1,\ldots,N\}, k \in \mathbb{K}_{ij}^p,\end{aligned} \tag{7.16}$$

$$\sum_{k \in K_{ij}} v_{ij}^{pk} = 1, \qquad i \in \{1,\ldots,N-1\}, \quad j \in \{i+1,\ldots,N\}, p \in \mathbb{Q}_{ij}, \tag{7.17}$$

$$\sum_{a \in A^+} v_a + \sum_{a \in A^-} (1 - v_a) \leq 2\eta, \tag{7.18}$$

$$L \leq \bar{L} - \theta, \tag{7.19}$$

$$x_i, y_i \in \mathbb{R}_+, \qquad i,j \in \{1,\ldots,N\}, \tag{7.20}$$

$$v_{ij}^{pk} \in \{0,1\}, \qquad i,j \in \{1,\ldots,N\}, \quad p \in \mathbb{Q}_{ij}, k \in \mathbb{K}_{ij}^p. \tag{7.21}$$

A cada passo da busca local, na estratégia de descida, a resolução do modelo é interrompida quando a primeira solução factível é encontrada ou quando o tempo limite de 300s é atingido. Por outro lado, na estratégia de máxima descida, a resolução do modelo é interrompida ao obter a solução ótima ou quando o tempo limite de 300s é atingido.

A terceira estratégia de busca, *Proximity Search*, é baseada na matheurística de refinamento proposta por Fischetti e Monaci (2014). Neste caso, foi utilizado o modelo (7.13) – (7.21) sem a restrição (7.18), que passa a ser considerada como função objetivo

$$\min \quad \sum_{a \in A^+} v_a + \sum_{a \in A^-} (1 - v_a) + M\gamma \tag{7.22}$$

e substituindo a restrição (7.19) por

$$L \leq \bar{L} - \theta + \gamma, \tag{7.23}$$

em que γ é uma variável real positiva, similar a uma variável de folga, que tem como objetivo permitir que a solução atual seja também factível para a vizinhança. A função objetivo (7.22) minimiza o número de arestas modificadas nas restrições de não sobreposição, i.e., minimiza a quantidade de alelos alterados no cromossomo. Seguindo a estratégia proposta por Fischetti e Monaci (2014), consideramos $\theta = 1$, atualizando para $\theta = 0,5$ e $\theta = 0,25$ caso não seja encontrada solução melhor. O tempo limite considerado para cada passo da busca foi limitado a 10s.

7.1.3.3 Testes preliminares

Em testes preliminares, consideramos quatro versões para as vizinhanças, variando a quantidade de arestas que podem ser alteradas (η) e o tamanho do passo (θ) de melhora exigido na função objetivo:

1ª) consideramos $\eta = 5$ (até 5 arestas podem ser trocadas na restrição de não sobreposição entre peças) e $\theta = 0,1$; caso nenhuma solução seja encontrada, tomamos $\eta = 10$; caso nenhuma solução seja encontrada novamente, finalizamos a busca nesta vizinhança;

2ª) consideramos $\eta = 5$ e $\theta = 0,5$; caso nenhuma solução seja encontrada, definimos $\theta = 0,25$; caso nenhuma solução seja encontrada novamente, atualizamos $\theta = 0,125$; e não encontrando solução, terminamos a busca;

3ª) sendo n o tamanho do cromossomo, consideramos $\eta = n/4$ e $\theta = 0,1$; caso nenhuma solução seja encontrada, atualizamos $\eta = n/2$; caso nenhum solução seja encontrada novamente, concluímos a busca;

4ª) por fim, consideramos $\eta = n/4$ e $\theta = 0,5$; caso nenhuma solução seja encontrada, atualizamos $\theta = 0,25$; caso nenhuma solução seja encontrada novamente, atualizamos $\theta = 0,125$; não encontrando solução, encerramos a busca.

Os resultados mostraram que a terceira opção combinada com a estratégia de máxima descida apresentou os melhores resultados. Em resumo, a matheurística desenvolvida combina o método BRKGA com uma busca local, chamamos o método de BRKGA-LS.

7.2 IRACE

Por se tratar de uma matheurística evolutiva, a BRKGA-LS possui vários parâmetros que influenciam sua eficiência. Esses parâmetros são apresentados na Tabela 9 juntamente com seus respectivos intervalos de valores satisfatórios, segundo Gonçalves e Resende (2011).

Tabela 9 – Parâmetros e valores recomendados.

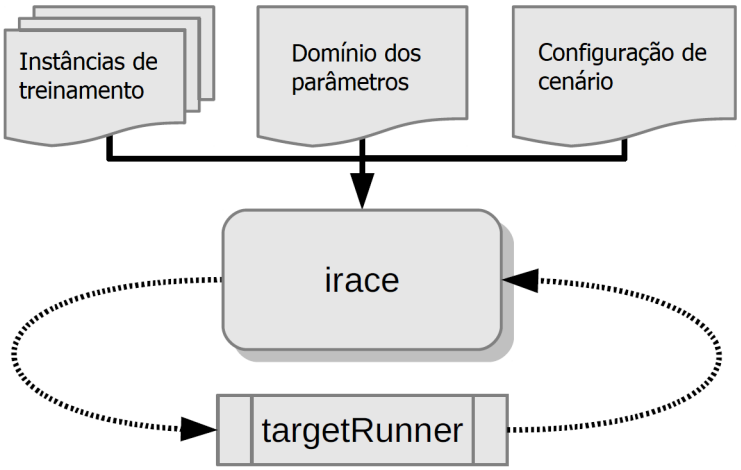
Parâmetros	**Valores recomendados**
tamanho p da população	$p = an$, sendo a uma constante, $1 \leq a \in \mathbb{R}$ e n o comprimento do cromossomo
tamanho pe da população elite	$0,10p \leq pe \leq 0,25p$
tamanho pm da população de mutantes	$0,10p \leq pm \leq 0,30p$
probabilidade ρa de herança da chave elite	$0,5 < \rho a \leq 0,8$

Em busca de uma boa configuração de parâmetros para a BRKGA-LS, utilizamos o pacote *irace*, desenvolvido por López-Ibáñez *et al.* (2016b), que dispõem de métodos estatísticos

para encontrar a(s) melhor(es) combinação(ões) de valores para um dado conjunto de parâmetros. O *irace* é um software livre que possui um procedimento de corrida iterada implementado e cujo principal objetivo é configurar automaticamente algoritmos de otimização, ou seja, encontrar a configuração mais apropriada para um algoritmo, dado um conjunto de instâncias. A versão mais recente, e também a utilizada nesta pesquisa, é a versão 2.4. Detalhes sobre a instalação do *irace* em diferentes sistemas operacionais podem ser encontrados em López-Ibáñez *et al.* (2016a) ou endereço eletrônico <http://iridia.ulb.ac.be/irace>.

Na Figura 38, apresentamos uma visão geral do pacote *irace*, que recebe como entrada as configurações que definem o cenário dos testes, os parâmetros do algoritmo de otimização que devem ser avaliados e seus respectivos intervalos de valores admissíveis. Em seguida, o *irace* procura por uma boa configuração para os parâmetros, executando o algoritmo de otimização para um conjunto de instâncias e configurações. O *targetRunner*, que pode ser uma função do R ou um executável, funciona como um intermediador entre o *irace* e o algoritmo de otimização. O *targetRunner* recebe uma instância e uma configuração de parâmetros do *irace*, passa essa informação para o algoritmo de otimização, que é executado e retorna para o *targetRunner* um valor representativo, por exemplo, um custo, e esse repassa essa informação ao *irace*.

Figura 38 – Fluxo das informações no *irace*.



Fonte: Adaptada de López-Ibáñez *et al.* (2016b).

7.2.1 Ajuste de parâmetros do BRKGA

Antes de executar o *irace* é necessário fornecer as seguintes informações:

- os parâmetros a serem ajustados e seus respectivos valores admissíveis;
- o *targetRunner*;
- um conjunto de instâncias de treinamento;

- as opções do *irace*.

Além dessas informações, existem ainda algumas outras opcionais como uma configuração inicial para o algoritmo de otimização ou configurações proibidas. Todas as informações são fornecidas como arquivos de texto ou como objetos do R. Com todas as informações disponíveis, o *irace* pode ser executado diretamente do R ou pela linha de comando **$IRACE-HOME/bin/irace –scenario scenario.txt**, sendo $IRACE-HOME o diretório de instalação do *irace*. A seguir, apresentamos um breve resumo sobre os itens. Para mais detalhes, veja López-Ibáñez *et al.* (2016a).

7.2.1.1 O targetRunner

É uma função do R ou um executável, disponível no pacote *irace*, que deve ser modificado de modo que o algoritmo de otimização possa ser executado. Como já mencionado, ele deve ser capaz de executar o algoritmo de otimização fornecendo a instância e os parâmetros especificados pelo *irace* e deve retornar um número real avaliativo da configuração testada. O código do *targetRunner* utilizado para configurar os parâmetros da matheurística encontra-se disponível no Apêndice A.

7.2.1.2 Os parâmetros a serem ajustados e seus respectivos intervalos

Os parâmetros são fornecidos ao *irace* através do arquivo de texto **parameters.txt**. Cada parâmetro deve ser associado a um tipo que define o seu domínio e o modo como o *irace* vai tratar com ele. Esses tipos são: “r”, “i”, “c” e “o”. Parâmetros que recebem números reais devem ser classificados como “r” e um intervalo do tipo “(<mínimo>,<máximo>)” deve ser especificado. O intervalo é fechado, ou seja, eventualmente os extremos podem ser usados. Parâmetros cujos valores são inteiros são classificados como “i”. Assim como para os parâmetros tipo “r”, também é necessário especificar um intervalo de valores. Os parâmetros do tipo “c”, chamados de categóricos, são aqueles que possuem um conjunto finito de valores, que devem ser especificados da seguinte maneira “(<valor 1>, ..., <valor n>)”. Por fim, os parâmetros ordinais, ou do tipo “o”, são definidos por um conjunto ordenado de possíveis valores no mesmo formato que os parâmetros categóricos. A descrição do espaço de parâmetros é dada como uma tabela dentro do arquivo de texto, em que cada linha define um parâmetro, da seguinte maneira: **<nome> <label> <tipo> <intervalo> [| <condição>]**. O arquivo “parameters.txt” utilizado para configurar os parâmetros da matheurística proposta encontra-se no Apêndice A.

7.2.1.3 As opções do irace e as instâncias de treinamento

As definições mais gerais são feitas no arquivo de texto **scenario.txt**. Nesse arquivo são definidos a localização dos diretórios de execução do *irace*, do arquivo de parâmetros e do *targetRunner*, o número máximo de experimentos e as casas decimais relevantes. Também é

definido nesse arquivo o conjunto de instâncias de treinamento, que pode ser todo um diretório (trainInstancesDir) ou algumas instâncias específicas (trainInstancesFile). Neste último caso, deve ser fornecido um arquivo texto em que cada linha possui o nome de uma instância. Novamente, o arquivo "scenario.txt" utilizado encontra-se no Apêndice A.

7.2.2 Configuração fornecida pelo irace

Para a execução do *irace* foi utilizado um computador com processador Intel Core i7-7700, 3.6 GHz com 16GB de RAM e sistema operacional Ubuntu 16.04. Com o *irace* buscamos ajustar os quatro parâmetros apresentados na Tabela 9 e a quantidade máxima de gerações para interromper o BRKGA. A melhor configuração encontrada é apresentada na Tabela 10, em que n é o tamanho do cromossomo. A saída completa do *irace* está detalhada na Apêndice A.

Tabela 10 – Parâmetros da matheurística obtidos pelo *irace*.

Parâmetros	**Configuração**
Tamanho da população	4 x n
Fração da população à ser considera elite	0,19
Fração da população à ser substituída por mutantes	0,24
Probabilidade do descendente herdar um alelo do pai elite	0,78
Número máximo de gerações	300

7.3 Experimentos computacionais

Nos experimentos computacionais foram utilizadas 24 instâncias da literatura do problema de corte e empacotamento, com um total de peças variando entre 3 e 35. Foram utilizadas instâncias com peças convexas e/ou não convexas e não foi considerada a possibilidade de rotação das peças. Mais detalhes, como o nome das instâncias, altura da placa, quantidade de tipos diferentes de peças e total de peças são apresentados na Tabela 11. As instâncias podem ser encontrados no endereço eletrônico do ESICUP (<https://www.euro-online.org/websites/esicup/>).

Durante os testes, cada instância foi resolvida 10 vezes com diferentes *seeds*. O modelo foi implementado em linguagem C/C++ utilizando o *software* de otimização ILOG CPLEX 12.6, com os parâmetros *default*, e os testes computacionais foram realizados em um computador Intel Core i7-7700 (3.60GHz x 8), com 16GB de memória RAM e sistema operacional Ubuntu 16.04. Foi utilizado o BRKGA API proposto por Toso e Resende (2012), que pode ser encontrado em: <http://mauricio.resende.info/src/brkgaAPI/>. Para todos os testes, os parâmetros para o BRKGA são os apresentados na Tabela 10, em que n é o tamanho do cromossomo.

Na Tabela 12, apresentamos os resultados do BRKGA-LS. A primeira coluna contém o nome das instâncias. As colunas 2 - 7 apresentam os resultados obtidos pelas 300 gerações do

Tabela 11 – Informações sobre as instâncias.

Instâncias	Placa	Peças	
	Altura	Tipos	Total
three	7	3	3
threep2	7	3	6
threep3	7	3	9
threep2w9	9	3	6
threep3w9	9	3	9
fu5	38	4	5
fu6	38	5	6
fu7	38	6	7
fu8	38	7	8
fu9	38	8	9
fu10	38	9	10
fu12	38	10	12
shapes2	40	4	8
shapes4	13	4	4
shapes8	20	4	8
rco1	15	7	7
rco2	15	7	14
rco3	15	7	21
rco4	15	7	28
rco5	15	7	35
blazewicz1	15	7	7
blazewicz2	15	7	14
blazewicz3	15	7	21
blazewicz4	15	7	28

BRKGA, sendo que as colunas 2 - 5 apresentam as médias dos resultados, mais especificamente as médias do comprimento (Layout), da iteração em que esse comprimento foi encontrado, do tempo necessário para encontrar esse comprimento e do tempo total para as 300 gerações, enquanto as colunas 6 e 7 apresentam o menor e o maior comprimento de empacotamento encontrados, L_{min} e L_{max}, respectivamente. As colunas 8 e 9 apresentam os resultados (em média) após a aplicação da heurística de melhoria, sendo o comprimento (Layout) e o tempo total (BRKGA-LS), respectivamente. As colunas 10 e 11 apresentam, respectivamente, o menor e o maior empacotamento (L_{min} e L_{max}) encontrados dentro das 10 execuções para cada instância e, por fim, a melhoria (em %) obtida ao aplicarmos a da busca local é reportada na última coluna. Os símbolos "-" indicam que não foram encontradas soluções factíveis dentro das 300 gerações.

Analisando a Tabela 12, observamos que em 300 gerações do BRKGA, a matheurística foi capaz de encontrar soluções factíveis para 22 das 24 instâncias avaliadas, sendo que em 3 casos (threep2w9, fu5 e fu7) a solução ótima foi encontrada em pelo menos uma das 10 rodadas (L_{min}). Aplicando a busca local à melhor solução obtida pelo BRKGA, todas as soluções foram melhoradas, em média 19%, com melhorias variando entre 3% e 45%. Soluções factíveis foram obtidas para as duas instâncias (shapes8 e blazewicz4) que não haviam sido resolvidas durante as gerações do BRKGA. Para 11 instâncias, a solução ótima é a solução média e em 16 instâncias a solução ótima foi encontrada em pelo menos uma das 10 rodadas (L_{min}).

Tabela 12 – Resultados obtidos pela matheurística proposta.

Instâncias	BRKGA				L_{min}	L_{max}	BRKGA-LS		L_{min}	L_{max}	Melhoria
	Layout	Iteração	Tempo (s)	T. Final (s)			Layout	T. Total (s)			(%)
three	6,2	85	1	2	6,2	6,2	**6,0**	2	**6,0**	**6,0**	3,2
threep2	10,3	69	3	11	9,7	11,0	**9,3**	13	**9,3**	**9,3**	9,7
threep3	15,9	97	13	39	14,6	17,0	13,6	644	**13,5**	13,7	14,7
threep2w9	8,5	72	3	11	**8,0**	9,0	**8,0**	15	**8,0**	**8,0**	6,3
threep3w9	13,3	152	20	39	11,5	15,0	**11,0**	828	**11,0**	11,1	17,0
fu5	18,6	40	1	7	**17,9**	20,0	**17,9**	8	**17,9**	**17,9**	3,6
fu6	24,8	84	3	11	24,0	28,0	23,5	11	**23,0**	24,0	5,2
fu7	25,4	67	4	17	**24,0**	28,0	**24,0**	17	**24,0**	**24,0**	5,3
fu8	28,6	148	12	25	25,0	34,0	**24,0**	26	**24,0**	**24,0**	16,0
fu9	33,9	94	12	38	29,0	38,0	25,5	60	**25,0**	26,0	24,8
fu10	37,4	142	25	51	34,0	40,0	29,1	579	**28,7**	30,0	22,2
fu12	46,1	138	46	98	42,0	52,0	33,9	1304	33,2	34,0	26,6
shapes2	25,6	176	393	643	20,0	30,0	**14,0**	782	**14,0**	**14,0**	45,2
shapes4	29,1	75	11	42	26,0	32,0	**24,0**	44	**24,0**	**24,0**	17,6
shapes8	-	-	-	659	-	-	29,0	1861	29,0	29,0	
rco1	9,3	103	6	18	9,0	10,5	**8,0**	19	**8,0**	**8,0**	13,8
rco2	21,7	149	82	162	19,0	27,0	15,4	1279	**15,0**	16,0	29,1
rco3	33,3	238	701	873	28,0	39,3	23,2	1870	22,4	24,0	30,3
rco4	42,6	279	3294	3498	40,0	48,5	33,5	4407	30,0	43,0	21,4
rco5	55,0	293	11148	11361	53,0	58,0	39,4	12356	38,1	41,3	28,3
blazewicz1	11,0	152	38	74	9,4	15,5	**7,4**	744	**7,4**	7,5	32,5
blazewicz2	21,4	266	1193	1327	19,0	25,0	15,8	2229	14,8	20,5	26,1
blazewicz3	32,6	148	5420	10964	29,5	35,9	27,0	11775	24,8	33,4	17,2
blazewicz4	-	-	-	59082	-	-	53,4	59983	50,1	56,1	
Média		133	975	3711				4202			18,9

O BRKGA encontrou soluções com, em média, 133 gerações e um tempo computacional de, em média, 975 segundos, enquanto a busca local necessitou de, em média, 492 segundos para melhorar as soluções. A matheurística proposta, com a aplicação da heurística de melhoramento, mostra-se robusta, pois conseguiu, para a maioria das instâncias testadas, valores máximos e mínimos de comprimento do empacotamento bem próximos aos valores médios, além de valores médios iguais ou bem próximos aos valores ótimos.

Comparamos a matheurística proposta (BRKGA-LS) com outra matheurística encontrada na literatura: a 3PM, desenvolvida por Cherri, Carravilla e Toledo (2016), que se baseia na estratégia *relax-and-fix*, utiliza o VND (do inglês *Variable Neighborhood Descent*) como heurística de melhoria e o modelo de Gomes e Oliveira (2006) para compactação das soluções. Para os testes computacionais foi utilizado o mesmo computador reportado por Cherri, Carravilla e Toledo (2016), um Intel Xeon E5-2620 2.00GHz, com 64GB de memória RAM e sistema operacional Ubuntu 12.04.

Na Tabela 13 são apresentados os resultados obtidos por ambas as matheurísticas. A primeira coluna contém o nome das instâncias. As colunas 2 e 3 apresentam os comprimentos e o tempo computacional da matheurística 3PM, as colunas 4 e 5 apresentam os comprimentos e o tempo computacional da matheurística BRKGA-LS e, por fim, a melhoria (em %) obtida pela BRKGA-LS é reportada na última coluna.

Analisando a Tabela 13, temos que a matheurística 3PM conseguiu solução factível para todas as instâncias em um tempo médio de 125 segundos, sendo que em 20 delas, a 3PM conseguiu a solução ótima. Comparando a BRKGA-LS com a 3PM concluímos que a BRKGA-LS é competitiva com a literatura, pois foi capaz de encontrar as mesmas soluções que 3PM para 11 instâncias e para outras 8, a solução da BRKGA-LS está a até 10% do ótimo.

Como esperado, os piores resultados são apresentados nas instâncias blazewicz, que possuem peças não convexas. Dado que optamos por utilizar a estratégia de transformar peças não convexas em diversas partes convexas, o NFP de peças não convexas é formado pela união dos NFP das partes convexas. Isso produz maior infactibilidade das soluções após a decodificação do cromossomo, uma vez que o cromossomo codifica qual aresta da parte p do NFP_{ij}^{p} é utilizada para garantir a não sobreposição entre as peças i e j, ou seja, esse método apresenta dificuldade em obter soluções viáveis para instâncias que possuem suas peças não convexas divididas em partes convexas.

Podemos concluir que a matheurística proposta é competitiva. Contudo, seu principal ponto positivo é ser facilmente adaptável a problemas biobjetivos, como o estudado no Capítulo 8.

Tabela 13 – Comparação entre a BRKGA-LS e a 3PM.

Instâncias	3PM		BRKGA-LS		GAP das soluções (%)
	Layout	Tempo (s)	Layout	Tempo (s)	
three	**6,0**	3	**6,0**	2	**0,0**
threep2	**9,3**	12	**9,3**	13	**0,0**
threep3	**13,5**	183	13,6	644	-0,5
threep2w9	**8,0**	36	**8,0**	15	**0,0**
threep3w9	**11,0**	191	**11,0**	828	**0,0**
fu5	**17,9**	1	**17,9**	8	**0,0**
fu6	**23,0**	32	23,5	11	-2,2
fu7	**24,0**	5	**24,0**	17	**0,0**
fu8	**24,0**	21	**24,0**	26	**0,0**
fu9	**25,0**	52	25,5	60	-2,0
fu10	**28,7**	266	29,1	579	-1,3
fu12	32,0	187	33,9	1304	-5,9
shapes2	**14,0**	8	**14,0**	782	**0,0**
shapes4	**24,0**	2	**24,0**	44	**0,0**
shapes8	**26,0**	187	29,0	1861	-11,5
rco1	**8,0**	44	**8,0**	19	**0,0**
rco2	**15,0**	255	15,4	1279	-2,7
rco3	**22,0**	265	23,2	1870	-5,6
rco4	**29,0**	83	33,5	4407	-15,4
rco5	36,7	210	39,4	12356	-7,4
blazewicz1	**7,4**	23	**7,4**	744	**0,0**
blazewicz2	14,0	69	15,8	2229	-13,1
blazewicz3	**20,5**	340	27,0	11775	-31,7
blazewicz4	27,9	518	53,4	59983	-91,3
Média		125		4202	

CAPÍTULO

8

MATHEURÍSTICA BASEADA NO BRKGA PARA O PROBLEMA INTEGRADO DE EMPACOTAMENTO DE PEÇAS IRREGULARES EM FAIXAS E DE CAMINHO MÍNIMO DE CORTE

Neste capítulo, é apresentada uma matheurística para o problema integrado de empacotamento de peças irregulares em faixas e de caminho mínimo de corte, considerando o empacotamento em domínio contínuo e o corte por arestas. Na Seção 8.1, é apresentado um modelo baseado no Problema do Carteiro Rural (RPP - do inglês *Rural Postman Problem*) que trata o CPDP com corte por arestas, desenvolvido em conjunto com o então aluno de mestrado Everton Fernandes da Silva. A matheurística baseada no BRKGA para o problema integrado é detalhada na Seção 8.2, enquanto que os experimentos computacionais são discutidos na Seção 8.3.

8.1 Modelo para o CPDP

O problema de determinação do caminho de corte (CPDP), como já mencionado, tem como um dos seus possíveis objetivos encontrar o menor percurso para cortar todas as peças em um dado empacotamento. Silva (2016) propõem dois modelos matemáticos para tratar o problema. O primeiro é baseado na formulação clássica do problema do carteiro rural (RPP - do inglês *Rural Postman Problem*) e o segundo é baseado no problema do caixeiro viajante (TSP - do inglês *Traveling Salesman Problem*). Para ambos os casos os autores consideram as seguintes características:

- o corte é feito a laser em uma placa metálica fina, logo, o tempo de perfuração é desprezado;
- a ferramenta de corte não se desgasta durante o corte, ou seja, não há alteração na sua capacidade de corte;
- o corte é feito por apenas uma ferramenta de corte;
- o diâmetro da ferramente é pequeno, assim não existe espaço significativo entre as peças após o corte; e
- a ferramenta de corte sempre inicia o percurso em uma origem pré-definida, chamada de depósito, e retorna a ela ao final do percurso.

Sobre as características da placa e das peças, é considerado que:

- as arestas das peças são independentes, ou seja, não é necessário que a peça seja inteiramente cortada de uma única vez, como acontece no corte por peça;
- não existe direção obrigatória para o corte das arestas;
- como a placa é fina, o tempo de perfuração pode ser ignorado; e
- a placa não é afetada pelo calor durante o corte.

O modelo baseado no RPP apresenta uma formulação simples, eficiente e obteve melhores resultados quando comparado ao modelo baseado no TSP. Nesse modelo, os vértices das peças a serem cortadas são representados pelos nós do RPP, enquanto os lados das peças correspondem às arestas obrigatórias. As demais arestas do grafo completo representam os possíveis movimentos da ferramenta de corte entre os vértices da peça, i.e., os movimentos aéreos (ou ociosos) da ferramenta. Se todos os vértices no grafo possuírem cardinalidade par e sem subciclos ilegais então a solução é garantidamente um caminho fechado. Os conjuntos, os parâmetros e as variáveis do modelo estão definidos a seguir.

Conjuntos e parâmetros

- $\mathbb{V}$: conjunto de vértices;
- $\mathbb{A}$: conjunto de arestas;
- $\mathbb{A}_{\mathbb{R}}$: conjunto de arestas obrigatórias ($\mathbb{A}_{\mathbb{R}} \subseteq \mathbb{A}$);
- $\mathbb{P}$: conjunto de componentes conexos no subgrafo definido por $\mathbb{A}_{\mathbb{R}}$;
- $\mathbb{V}_{k_1}$: conjunto de vértices da componente $k_1 \in \mathbb{P}$;
- $\delta(i)$: conjunto de arestas incidentes no vértice $i \in \mathbb{V}$;

- c_e: custo de percorrer a aresta $e \in \mathbb{A}$.

Variáveis

- x_e: número de vezes que a aresta e é percorrida;
- w_i: variável auxiliar utilizada para manter a cardinalidade par do vértice i.

O modelo para a determinação do caminho de corte baseado no RPP, utilizado por Silva (2016), é dado por:

$$\min \sum_{e \in \mathbb{A}} c_e x_e \tag{8.1}$$

$$s.a. |\mathbb{A}_{\mathbb{R}} \cap \delta(i)| + \sum_{e \in \delta(i)} x_e = 2w_i, \qquad \forall i \in \mathbb{V}, \tag{8.2}$$

$$\sum_{i \in \mathbb{V}_{k_1}} \sum_{k_2 \in \mathbb{P} \setminus \{k_1\}} \sum_{j \in \mathbb{V}_{k_2}} \sum_{e \in \delta(i) \cap \delta(j)} x_e \geq 2, \qquad \forall k_1 \in \mathbb{P}, \tag{8.3}$$

$$x_e \in \mathbb{Z}_+, \qquad \forall e \in \mathbb{A}, \tag{8.4}$$

$$w_i \in \mathbb{Z}_+, \qquad \forall i \in \mathbb{V}. \tag{8.5}$$

O modelo (8.1) – (8.5) visa minimizar o custo (distância) de percorrer o caminho de corte. As restrições (8.2) garantem que cada vértice tenha cardinalidade par, em que $|\mathbb{A}_{\mathbb{R}} \cap \delta(i)|$ corresponde ao número de arestas obrigatórias incidentes em i. As restrições (8.3) garantem que não exista subciclo ilegal, ou seja, cada subconjunto de vértices k_1 ($k_1 \in P$) conectados às arestas obrigatórias deve ter no mínimo duas arestas conectando-os ao resto do grafo. Por fim, as restrições (8.4) e (8.5) definem o domínio das variáveis do modelo.

8.2 Matheurística para o problema integrado de empacotamento de peças irregulares em faixas e de caminho mínimo de corte

A matheurística baseada no BRKGA, proposta para resolver o problema integrado de empacotamento de peças irregulares em faixas e de caminho mínimo de corte, possui como base a matheurística apresentada na Seção 7.1, agora com dois modelos a serem resolvidos no decodificador. Em outras palavras, temos que para cada cromossomo, o decodificador buscará o melhor empacotamento possível para as peças, resolvendo o modelo (7.1) – (7.7) apresentado na Subseção 7.1.2 e, caso o empacotamento resultante não possua sobreposição, buscará então o melhor caminho de corte para esse empacotamento através do modelo (8.1) – (8.5), apresentado na Seção 8.1.

Com essa estratégia, cada cromossomo possui dois valores representativos, ou *fitness*: os comprimentos do empacotamento e do caminho de corte. Contudo, o BRKGA utiliza uma única *fitness* na evolução dos cromossomos. Na literatura, os trabalhos de Roque, Fontes e Fontes (2017a), Roque, Fontes e Fontes (2017b) e Tangpattanakul, Jozefowiez e Lopez (2013) utilizam o BRKGA para problemas multiobjetivos. Nesses trabalhos, os autores calculam a *fitness* dos cromossomos utilizando a estratégia de classificação não-dominada (*fast non dominated sorting approach*) e a distância de aglomeração (*crowding distance*) propostas ao *Nondominated Sorting Genetic Algorithm II* (NSGA-II), proposta por Deb *et al.* (2002).

Na abordagem de classificação não-dominada, as soluções são classificadas de acordo com as fronteiras de dominância, ou seja, as soluções não dominadas formam a primeira fronteira, as soluções não dominadas desconsiderando a primeira fronteira, formam a segunda fronteira, e assim sucessivamente. Enquanto que a distância de aglomeração visa preservar a diversidade da população dentro de cada fronteira. Nos Algoritmos 1 e 2 estão descritos, respectivamente, os passos da estratégia de classificação não-dominada e da distância de aglomeração.

Algoritmo 1 – Estratégia de classificação não-dominada

1: **procedimento** FAST-NON-DOMINATED-SORT(Pop)
2: **para cada** $p \in Pop$
3: $S_p = \emptyset$
4: $n_p = 0$
5: **para cada** $q \in Pop$
6: **se** $(p \prec q)$ **então** ▷ Se p domina q
7: $S_p = S_p \cup \{q\}$ ▷ q é adicionado ao conjunto de soluções dominadas por p
8: **senão**
9: **se** $(q \prec p)$ **então**
10: $n_p = n_p + 1$ ▷ Adicionar 1 ao contador de dominação de p
11: **se** $(n_p = 0)$ **então** ▷ p pertence a primeira fronteira
12: $p_{rank} = 1$
13: $F_1 = F_1 \cup \{p\}$
14: $i = 1$ ▷ Inicializa o contador de fronteira
15: **enquanto** $F_i \neq \emptyset$ **faça**
16: $Q = \emptyset$ ▷ Usado para guardar os membros da próxima fronteira
17: **para cada** $p \in F_i$
18: **para cada** $q \in S_p$
19: $n_q = n_q - 1$
20: **se** $(n_q = 0)$ **então** ▷ q pertence à próxima fronteira
21: $q_{rank} = i + 1$
22: $Q = Q \cup \{q\}$
23: $i = i + 1$
24: $F_i = Q$

Assim, utilizando a estratégia de classificação não-dominada e a distância de aglomeração, o valor de *fitness* de um cromossomo com duas funções objetivos passa a ser um único

Algoritmo 2 – Distância de aglomeração

1: **procedimento** CROWDING-DISTANCE($Front$)
2: $l = |Front|$ ▷ Número de soluções na Fronteira
3: **para cada** i, $Dist_{Front[i]} = 0$ ▷ Inicializa a Distancia
4: **para cada** objetivo m
5: $Front = \text{sort}(Front, m)$ ▷ Ordena usando cada valor da função objetivo
6: $Dist_{Front[1]} = Dist_{Front[l]} = \infty$ ▷ Assim, os limitantes são sempre selecionados
7: **para** $i = 2$ até $(l-1)$ **faça** ▷ Para todos os outros pontos
8: $Dist_{Front[i]} = Dist_{Front[i]} + (Front[i+1].m - Front[i-1].m)/(f_m^{max} - f_m^{min})$

valor e o BRKGA, que antes retornava apenas a melhor solução elite, agora retorna a primeira fronteira de soluções.

8.3 Experimentos computacionais

Os testes computacionais foram divididos em duas fases. Na primeira fase, apresentada na Subseção 8.3.1, comparamos os resultados obtidos pelo BRKGAIntegrado e pelo MIEPICCP-VL biobjetivo. Na segunda fase, apresentada na Subseção 8.3.2, discutimos os resultados do BRKGAIntegrado na resolução de instâncias maiores.

Durante os testes, cada instância foi resolvida 10 vezes com diferentes sementes (*seeds*). Os modelos foram implementados em linguagem C/C++ utilizando o *software* de otimização ILOG CPLEX 12.6, com os parâmetros *default*, e os testes computacionais foram realizados em um computador Intel Core i7-7700 (3.60GHz x 8), com 16GB de memória RAM e sistema operacional Ubuntu 16.04. Novamente, utilizamos o BRKGA API proposto por Toso e Resende (2012), que pode ser encontrado em: <http://mauricio.resende.info/src/brkgaAPI/>. Os parâmetros considerados para o BRKGA são apresentados na Tabela 14, em que n é o tamanho do cromossomo. Esses parâmetros foram baseados nos parâmetros utilizados na BRKGA-LS e nos trabalhos de Roque, Fontes e Fontes (2017a), Roque, Fontes e Fontes (2017b) e Tangpattanakul, Jozefowiez e Lopez (2013).

Tabela 14 – Parâmetros da matheurística.

Parâmetros	**Configuração**
Tamanho da população	5 x n
Fração da população à ser considera elite	0,20
Fração da população à ser substituída por mutantes	0,40
Probabilidade do descendente herdar um alelo do pai elite	0,70
Número máximo de gerações	300

8.3.1 Fase I - Comparação com MIEPICCP-VL

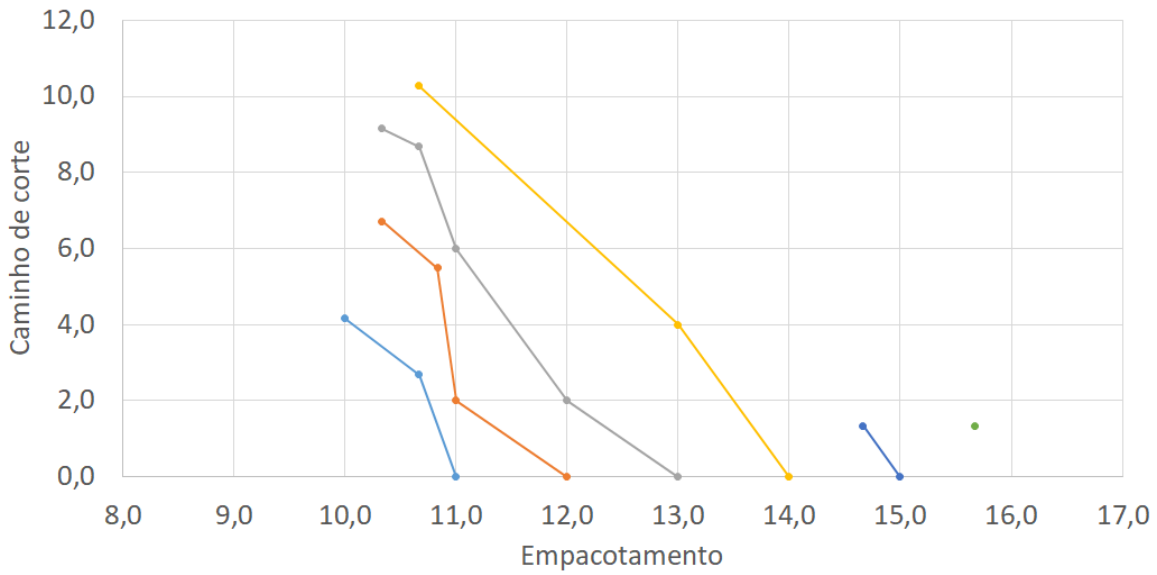
Para os experimentos computacionais dessa primeira fase, consideramos as mesmas 5 instâncias da literatura do problema de corte e empacotamento utilizadas nos testes com MIEPICCP-VL, apresentados na Subseção 6.2.2. Visando facilitar a leitura, detalhes como o nome das instâncias, altura da placa, quantidade de tipos diferentes de peças e total de peças são apresentados novamente na Tabela 15.

Tabela 15 – Informações sobre as instâncias.

Instâncias	Placa	Peças	
	Altura	Tipos	Total
three	7	3	3
threep2	7	3	6
threep3	7	3	9
shapes4	13	4	4
shapes4N	20	4	4

Dado que o BRKGAIntegrado retorna a primeira fronteira de soluções, i.e., todas as soluções não dominadas, agrupamos os resultados obtidos nas 10 rodadas e, utilizando o Algoritmo 1, reavaliamos as soluções e geramos as fronteiras novamente, conforme ilustrado na Figura 39, em que cada cor representa uma fronteira de soluções. Os gráficos reportando as mesmas informações para as demais instâncias estão disponíveis no Apêndice B.

Figura 39 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância three2.



Fonte: Elaborada pela autora.

Os valores das soluções pertencentes à primeira fronteira de cada instância são reportados na Tabela 16, em que comparamos com os resultados obtidos pelo MIEPICCP-VL biobjetivo, apresentados na Tabela 7 e reportados novamente a fim de facilitar a leitura. A primeira coluna

contém o nome das instâncias. As colunas 2 - 6 apresentam os resultados obtidos MIEPICCP-VL biobjetivo, sendo: o valor de ε utilizado, os comprimentos do empacotamento (layout) e do caminho de corte (CC), respectivamente, o GAP e o tempo computacional. O tempo limite (TL) nesse caso foi de 18000 segundos. As colunas 7 - 11 apresentam as soluções da primeira fronteira encontradas pelo BRKGAIntegrado: os comprimentos do empacotamento (layout) e do caminho de corte (CC), o tempo médio, mínimo e máximo considerando as 10 rodadas.

Tabela 16 – Comparação entre os resultados do MIEPICCP-VL biobjetivo e a primeira fronteira do BRKGAIntegrado.

Instâncias	**MIEPICCP-VL**					**BRKGAIntegrado**				
	ε	**Layout**	**CC**	**Gap (%)**	**Tempo (s)**	**Layout**	**CC**	**T. médio (s)**	**T. min. (s)**	**T. max. (s)**
three	6	6	9,8	0,0	2	6,2	2,3	8	7	9
	7	7	8,5	0,0	55	6,7	1,3			
						7,0	0,0			
threep2	10	10	16,6	38,4	TL	10,0	4,2	25	19	36
	11	11	15,4	49,4	TL	10,7	2,7			
						11,0	0,0			
threep3	14	14	23,4	47,9	TL	15,5	10,1	82	49	126
	15	15	25,5	71,7	TL	15,7	9,3			
	16	16	25,9	78,1	TL	15,8	7,3			
						16,3	7,1			
						16,7	3,3			
						17,7	0,7			
						18,0	0,0			
shapes4	24	24	27,3	57,0	TL	24,0	18,1	40	33	53
shapes4N	14	14	28,5	0,0	5843	16,0	27,7	39	31	48
	15	14	28,6	67,6	TL	17,0	23,7			
	16	16	23,9	74,4	TL	18,0	22,3			
	17	17	29,7	90,6	TL	19,8	21,8			
	18	18	25,8	86,1	TL	20,0	20,6			
	19	19	24,4	83,0	TL	22,0	20,1			
	20	19	29,1	89,7	TL	23,0	18,5			
	21	19	27,9	88,6	TL					
	22	20	22,7	81,8	TL					
	23	20	24,4	82,9	TL					
	24	24	24,7	83,2	TL					

Vale a pena ressaltar que o MIEPICCP-VL considera o empacotamento discreto e o corte por peças a partir de um vértice qualquer, enquanto que o BRKGAIntegrado considera o empacotamento em domínio contínuo e o corte por arestas. Contudo, os empacotamentos encontrados pelo MIEPICCP-VL também são empacotamentos factíveis ao BRKGAIntegrado, enquanto que os comprimentos dos caminhos de corte do MIEPICCP-VL podem ser utilizados como limitantes na comparação com os comprimentos obtidos pelo BRKGAIntegrado.

Analisando a Tabela 16, observamos que BRKGAIntegrado e o MIEPICCP-VL obtiveram um número semelhante de soluções não dominadas para cada instância. Como esperado,

essas soluções não são exclusivamente discretas, dado que o modelo permite o posicionamento das peças em domínio contínuo. Os resultados mostram que existe uma certa dificuldade para o BRKGAIntegrado obter soluções próximas às soluções ótimas do empacotamento, pois, com exceção da instância shapes4, a solução de empacotamento mínimo não foi obtida para nenhuma outra instância.

Em relação ao tempo computacional, observamos que o BRKGAIntegrado é uma matheurística rápida, conseguindo obter soluções para as instâncias em no máximo 82 segundos, na média. Com relação ao caminho de corte, como esperado, a matheurística resultou em caminhos significativamente menores, uma vez que o corte é realizado por aresta e não por peça. Destacamos também que para três soluções o corte foi realizado sem demandar movimento aéreo, ou seja, CC = $0,0$ (three - terceira solução; threep2 - terceira solução; e threep3 - sétima solução). Por fim, adicionar o modelo (8.1) – (8.5) para geração do caminho de corte resultou, para essas instâncias, em um gasto médio de 15 segundos a mais quando comparado ao tempo final do algoritmo BRKGA da matheurística BRKGA-LS.

8.3.2 Fase II - Experimentos com instâncias maiores

Dado o bom desempenho do BRKGAIntegrado para as instâncias pequenas para os testes da Fase I, na Fase II avaliamos o desemprenho do BRKGAIntegrado para instâncias maiores. Foram escolhidas as instâncias que possuem tempo computacional menor que 100s durante as 300 gerações do BRKGA, reportado na Tabela 12. Visando facilitar a leitura, detalhes, como o nome das instâncias, altura da placa, quantidade de tipos diferentes de peças e total de peças são novamente apresentados na Tabela 17.

Tabela 17 – Informações sobre as instâncias.

Instâncias	**Placa**	**Peças**	
	Altura	**Tipos**	**Total**
threep2w9	9	3	6
threep3w9	9	3	9
fu5	38	4	5
fu6	38	5	6
fu7	38	6	7
fu8	38	7	8
fu9	38	8	9
fu10	38	9	10
fu12	38	10	12
rco1	15	7	7
blazewicz1	15	7	7

Agrupamos os resultados obtidos nas 10 rodadas e utilizamos o Algoritmo 1 para reavaliar as soluções e gerar as fronteiras. Os gráficos reportando os detalhes das soluções factíveis e as fronteiras para cada instância estão disponíveis no Apêndice B. Na Tabela 18 apresentamos as soluções da primeira fronteira encontrada pelo BRKGAIntegrado, sendo: os comprimentos

do empacotamento (layout) e do caminho de corte (CC), o tempo médio, mínimo e máximo, considerando as 10 rodadas.

Analisando a Tabela 18, observamos que o BRKGAIntegrado foi capaz de obter soluções para instâncias que o MIEPICCP-VL não tinha sido capaz. No pior caso, o tempo médio foi de 1350 segundos (fu12). Novamente, o BRKGAIntegrado apresentou dificuldades em conseguir soluções próximas às soluções ótimas do empacotamento. As exceções estão nas instâncias threep2w9, fu5, fu7 e rco1, em que o BRKGAIntegrado encontrou soluções com empacotamento mínimo. Adicionar o modelo (8.1) – (8.5) para geração do caminho de corte resultou, para essas instâncias, em um gasto médio de 184 segundos a mais, quando comparado ao tempo final do algoritmo BRKGA da matheurística BRKGA-LS.

Concluímos que a matheurística proposta é promissora e consegue obter soluções para o problema integrado de empacotamento de peças irregulares em domínio contínuo e caminho mínimo de corte por arestas. Assim como para o BRKGA-LS, durante o processo de evolução muitas soluções infactíveis com relação ao empacotamento das peças são encontradas, logo a resolução do problema integrado não resulta em um número expressivo de soluções na primeira fronteira. Neste sentido, em trabalhos futuros, pretendemos explorar diferentes formas de decodificar a solução, como heurísticas tipo *Bottom-Left* (BL), e assim substituir o modelo (7.1) – (7.7) por uma heurística.

Tabela 18 – Resultados da primeira fronteira do BRKGAIntegrado aplicados a instâncias maiores.

Instâncias	**Layout**	**CC**	**T. médio (s)**	**T. min. (s)**	**T. max. (s)**
threep2w9	8,0	5,4	44	30	73
	8,7	3,1			
	9,0	0,0			
threep3w9	13,1	7,3	97	69	133
	13,5	7,2			
	14,3	4,4			
	15,3	2,1			
	16,0	2,0			
	17,0	0,0			
fu5	17,9	4,4	35	29	50
	24,0	0,0			
fu6	24,0	3,1	35	31	41
	28,0	0,0			
fu7	24,0	7,2	56	48	61
	27,0	6,8			
	28,0	0,0			
fu8	28,0	4,8	68	57	80
	34,0	2,0			
	38,0	0,0			
fu9	29,0	18,0	99	57	153
	34,0	0,0			
fu10	36,4	10,4	167	132	201
	38,0	1,6			
	48,0	0,0			
fu12	43,0	42,9	1350	530	2536
	43,5	36,2			
	45,2	35,8			
	45,2	35,8			
	46,1	34,4			
	46,3	34,3			
	46,5	23,6			
	46,6	14,8			
	48,0	9,6			
	49,8	6,9			
	50,4	5,7			
	51,1	3,6			
	52,0	2,8			
	53,3	2,7			
	62,0	0,0			
rco1	8,0	2,8	114	80	133
	9,0	0,0			
blazewicz1	9,0	11,4	196	104	251
	9,6	7,0			
	11,0	3,3			

CAPÍTULO

9

CONCLUSÕES E PESQUISAS FUTURAS

O problema de empacotamento aparece em diversos processos produtivos em que objetos maiores (placas) devem ser cortados em objetos menores (peças). O *Problema de Empacotamento de Peças Irregulares em Faixas*, estudado nesse trabalho, consiste em agrupar peças de formato irregular em uma placa de altura fixa e comprimento ilimitado, com o objetivo de minimizar o comprimento utilizado da placa. Uma vez definido o empacotamento das peças, surge, em algumas indústrias, a questão de encontrar o melhor modo de cortar as peças, ou seja, o *Problema de Determinar um Caminho de Corte* a ser seguido para executar o empacotamento anteriormente definido. O objetivo pode ser, por exemplo, reduzir o tempo total de corte, e assim possibilitar a execução de um maior número de placas ao final de um turno de trabalho.

Manber e Israni (1984), Imahori *et al.* (2008), Sherif, Jawahar e Balamurali (2014) e Anand e Babu (2015) destacam que as decisões tomadas na escolha do empacotamento influenciam fortemente a resolução do problema de caminho mínimo de corte, no entanto, é de nosso conhecimento que, até o momento, apenas Anand e Babu (2015) abordaram estes problemas de maneira integrada, porém apenas com peças regulares (retangulares). Neste trabalho, investigamos se é vantajoso integrar os problemas de empacotamento e de determinação de caminho de corte de peças irregulares e propomos algumas estratégias de integração.

Primeiramente, consideramos o caso em que o corte é feito por peça, ou seja, uma peça deve ser completamente cortada para que o corte da próxima peça seja iniciado. Para esse problema, foi proposta uma modelagem discreta para o empacotamento das peças, em que o corte por peça é feito a partir de um ponto pré-determinado, ou seja, cada peça é representada por um ponto fixo pelo qual a ferramenta de corte deve iniciar e concluir o corte. Em seguida, este modelo foi estendido a fim de analisar um maior número de possibilidades para o início do corte, ou seja, o modelo permite que todos os vértices da peça sejam alternativas para o início do corte da peça e escolhe o vértice mais vantajoso como ponto de entrada/saída da ferramenta de corte. Os resultados obtidos mostraram que é vantajoso integrar os problemas de empacotamento e de

caminho mínimo de corte, contudo como ambos problemas são de difícil resolução, o problema integrado é pelo menos tão difícil quanto os problemas isolados. Isso pode ser observado pelo aumento do tempo necessário para resolver o modelo a medida que as instâncias aumentam de tamanho.

No segundo caso estudado, as peças podem ser cortadas por arestas, ou seja, as arestas das peças podem ser cortadas de forma independente uma das outras, não havendo a necessidade de cortar todas as arestas de uma peça antes que o corte de uma outra peça seja iniciado. Primeiramente desenvolvemos uma matheurística baseada no BRKGA para o problema de empacotamento de peças irregulares em faixa. No BRKGA, cada solução é representada por um vetor de n chaves aleatórias. A transformação deste vetor em uma solução do problema é feita por um decodificador, que também retorna um valor representativo da solução decodificada. Na matheurística proposta, o decodificador é um modelo de programação linear que minimiza o comprimento da placa e a sobreposição entre as peças. Nesse modelo, as restrições de não sobreposição são escritas baseadas nas arestas dos *no-fit polygons* (NFP) entre os pares de peças. Ao final do algoritmo do BRKGA, aplicamos uma heurística de melhoramento à melhor solução elite encontrada. Os resultados obtidos foram próximos aos resultados ótimos disponíveis na literatura.

Por fim, com base nessa matheurística desenvolvida, propusemos uma matheurística para o problema integrado. O método incorpora o modelo de determinação de caminho de corte por aresta, proposto por Silva (2016). Os resultados se mostraram promissores, tendo a matheurística sido capaz de obter soluções para instâncias do problema integrado que não haviam sido resolvidas através dos modelos previamente propostos.

Resumindo, duas versões para o problema integrado foram consideradas neste trabalho. Na primeira, o problema é considerado discreto, ou seja, o empacotamento das peças é realizado em um objeto definido por uma malha e o corte é realizado por peça, a partir de um de seu vértices. Para esta versão foram propostas dois modelos matemáticos discretos. Na segunda versão, os problemas são tratados de forma contínua, isto é, o posicionamento das peças no objeto é livre e as peças podem ser cortadas por suas arestas. Neste caso, propusemos uma matheurística baseada no BRKGA.

Como trabalhos futuros, sugere-se melhorar as restrições dos modelos, por exemplo as restrições de empacotamento, como estudado por Rodrigues e Toledo (2017), e utilizar métodos para sua resolução, como clássicos de otimização multiobjetivo, métodos de decomposição ou adaptar o BRKGAIntegrado utilizando a estratégia *no-fit raster* utilizada em Mundim, Andretta e Queiroz (2017). Visando melhorar o BRKGAIntegrado, sugere-se adaptar e aplicar a heurística de melhoria proposta neste trabalho a cada uma das soluções de fronteira encontrada pelo método e, possivelmente, utilizar paralelismo a fim de tornar o método mais competitivo. Além disso, como já mencionado, pretendemos utilizar uma nova representação para o cromossomo, em que o vetor de chaves aleatórias codificaria a sequência e as rotações das peças, como em Júnior,

Pinheiro e Coelho (2017), substituir o modelo no decodificador por uma heurística baseada na *Bottom-Left* (BL) e comparar com o BRKGAIntegrado proposto nesse trabalho.

Por fim, considerando também a publicação de um método que integra os problemas de empacotamento de peças regulares e de determinação de caminho de corte por Anand e Babu (2015), pretende-se também adaptar o modelo integrado e/ou a matheurística proposta, considerando peças regulares, por exemplo, a envoltória retangular das peças irregulares, e comparar os resultados obtidos com os publicados.

REFERÊNCIAS

ALBANO, A.; SAPUPPO, G. Optimal allocation of two-dimensional irregular shapes using heuristic search methods. **IEEE Transactions on Systems, Man and Cybernetics**, v. 10, n. 5, p. 242 – 248, 1980. Citado na página 31.

ÁLVAREZ-VALDÉS, R.; CARRAVILLA, M. A.; OLIVEIRA, J. F. Cutting and packing. In: **Handbook of Heuristics**. [S.l.]: Springer International Publishing, 2018. p. 1 – 46. Citado na página 31.

ÁLVAREZ-VALDÉS, R.; MARTINEZ, A.; TAMARIT, J. A branch & bound algorithm for cutting and packing irregularly shaped pieces. **International Journal of Production Economics**, v. 145, n. 2, p. 463 – 477, 2013. Citado nas páginas 33 e 34.

ANAND, K. V.; BABU, A. R. Heuristic and genetic approach for nesting of two-dimensional rectangular shaped parts with common cutting edge concept for laser cutting and profile blanking processes. **Computers & Industrial Engineering**, v. 80, p. 111 – 124, 2015. Citado nas páginas 24, 40, 47, 48, 50, 51, 52, 111 e 113.

ART, R. C. **An approach to the two dimensional irregular cutting stock problem**. Cambridge, Massachusetts, USA, 1966. Citado na página 31.

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**, v. 33, n. 12, p. 879 – 891, 2001. Citado na página 31.

BENNELL, J. A.; DOWSLAND, K. A. Hybridising tabu search with optimisation techniques for irregular stock cutting. **Management Science**, v. 47, n. 8, p. 1160 – 1172, 2001. Citado na página 31.

BENNELL, J. A.; OLIVEIRA, J. F. The geometry of nesting problems: A tutorial. **European Journal of Operational Research**, v. 184, n. 2, p. 397 – 415, 2008. Citado nas páginas 27, 28, 29 e 30.

____. A tutorial in irregular shape packing problems. **Journal of the Operational Research Society**, v. 60, n. 1, p. 93 – 105, 2009. Citado na página 31.

BENNELL, J. A.; SONG, X. A comprehensive and robust procedure for obtaining the no-fit-polygon using minkowski sums. **Computers and Operations Research**, v. 35, n. 1, p. 267 – 281, 2008. Citado na página 31.

BURKE, E. K.; HELLIER, R.; KENDALL, G.; WHITWELL, G. A new bottom-left-fill heuristic algorithm for the two-dimensional irregular packing problem. **Operations Research**, v. 54, n. 3, p. 587 – 601, 2006. Citado na página 31.

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 32.

CHERRI, L. H.; CARRAVILLA, M. A.; TOLEDO, F. M. B. A model-based heuristic for the irregular strip packing problem. **Pesquisa Operacional**, v. 36, n. 3, p. 447 – 468, 2016. Citado na página 98.

CHERRI, L. H.; CHERRI, A. C.; CARRAVILLA, M. A.; OLIVEIRA, J. F.; TOLEDO, F. M. B.; VIANNA, A. C. G. An innovative data structure to handle the geometry of nesting problems. **International Journal of Production Research**, v. 56, n. 23, p. 7085 – 7102, 2018. 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**, v. 253, n. 3, p. 570 – 583, 2016. Citado nas páginas 29, 35, 86 e 87.

DANTZIG, G.; FULKERSON, R.; JOHNSON, S. Solutions of a large-scale traveling-salesman problem. **Journal of the Operations Research Society of America**, v. 2, p. 363 – 410, 1954. Citado na página 58.

DEB, K. Multi-objective optimization. In: **Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques**. [S.l.]: Springer US, 2014. p. 403 – 449. Citado nas páginas 56 e 58.

DEB, K.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, v. 6, n. 2, p. 182 – 197, 2002. Citado na página 104.

DEWIL, R.; VANSTEENWEGEN, P.; CATTRYSSE, D. Cutting path optimization using tabu search. **Key Engineering Materials**, v. 473, p. 739 – 748, 2011. Citado nas páginas 38, 39, 40, 41, 43 e 44.

____. Construction heuristics for generating tool paths for laser cutters. **International Journal of Production Research**, v. 52, n. 20, p. 5965 – 5984, 2014. Citado nas páginas 38, 39, 41, 42 e 44.

____. Sheet metal laser cutting tool path generation: Dealing with overlooked problem aspects. **Key Engineering Materials**, v. 639, p. 517 – 524, 2015. Citado nas páginas 44 e 45.

DEWIL, R.; VANSTEENWEGEN, P.; CATTRYSSE, D.; LAGUNA, M.; VOSSEN, T. An improvement heuristic framework for the laser cutting tool path problem. **International Journal of Production Research**, v. 53, n. 6, p. 1761 – 1776, 2015. Citado nas páginas 15, 44 e 45.

DOWSLAND, K. A.; DOWSLAND, W. B.; BENNELL, J. A. Jostle for position: Local improvement for irregular cutting patterns. **The Journal of the Operational Research Society**, v. 49, n. 6, p. 647 – 658, 1998. Citado na página 31.

DOWSLAND, K. A.; VAID, S.; DOWSLAND, W. B. An algorithm for polygon placement using a bottom-left strategy. **European Journal of Operational Research**, v. 141, n. 2, p. 371 – 381, 2002. Citado na página 31.

ELKERAN, A. A new approach for sheet nesting problem using guided cuckoo search and pairwise clustering. **European Journal of Operational Research**, v. 231, n. 3, p. 757 – 769, 2013. Citado na página 31.

FISCHETTI, M.; LUZZI, I. Mixed-integer programming models for nesting problems. **Journal of Heuristics**, v. 15, n. 3, p. 201 – 226, 2009. Citado nas páginas 32 e 33.

FISCHETTI, M.; MONACI, M. Proximity search for 0-1 mixed-integer convex programming. **Journal of Heuristics**, v. 20, n. 6, p. 709 – 731, 2014. Citado na página 91.

FOWLER, R. J.; PATERSON, M. S.; TANIMOTO, S. L. Optimal packing and covering in the plane are NP-complete. **Information Processing Letters**, v. 12, n. 3, p. 133 – 137, 1981. Citado nas páginas 23 e 31.

GOLDEN, B. L.; WONG, R. T. Capacitated arc routing problems. **Networks - Wiley Subscription Services, Inc., A Wiley Company**, v. 11, n. 3, p. 305 – 315, 1981. Citado na página 46.

GOMES, A. M.; OLIVEIRA, J. F. A 2-exchange heuristic for nesting problems. **European Journal of Operational Research**, v. 141, n. 2, p. 359 – 370, 2002. Citado na página 31.

____. Solving irregular strip packing problems by hybridising simulated annealing and linear programming. **European Journal of Operational Research**, v. 171, n. 3, p. 811 – 829, 2006. Citado nas páginas 32 e 98.

GONÇALVES, J. F.; RESENDE, M. G. C. Biased random-key genetic algorithms for combinatorial optimization. **Journal of Heuristics**, v. 17, n. 5, p. 487 – 525, 2011. Citado nas páginas 83, 84, 85 e 92.

HAN, G. C.; NA, S. J. A study on torch path planning in laser cutting processes part 2: Cutting path optimization using simulated annealing. **Journal of Manufacturing Systems**, v. 18, n. 2, Supplement 1, p. 62 – 70, 1999. Citado nas páginas 39 e 42.

IMAHORI, S.; KUSHIYA, M.; NAKASHIMA, T.; SUGIHARA, K. Generation of cutter paths for hard material in wire EDM. **Journal of Materials Processing Technology**, v. 206, n. 1, p. 453 – 461, 2008. Citado nas páginas 24, 39, 40, 43 e 111.

JAKOBS, S. On genetic algorithms for the packing of polygons. **European Journal of Operational Research**, v. 88, n. 1, p. 165 – 181, 1996. Citado na página 31.

JÚNIOR, B. A.; PINHEIRO, P. R.; COELHO, P. V. A parallel biased random-key genetic algorithm with multiple populations applied to irregular strip packing problems. **Mathematical Problems in Engineering**, v. 2017, p. 1 – 11, 2017. Citado nas páginas 32 e 113.

LEÃO, 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**, v. 54, n. 3, p. 712 – 721, 2016. Citado na página 35.

LEE, M. K.; KWON, K. B. Cutting path optimization in CNC cutting processes using a two-step genetic algorithm. **International Journal of Production Research**, v. 44, n. 24, p. 5307 – 5326, 2006. Citado nas páginas 41 e 43.

LÓPEZ-IBÁÑEZ, M.; CÁCERES, L. P.; DUBOIS-LACOSTE, J.; BIRATTARI, M. The irace package: user guide. **Technical Report TR/IRIDIA/2016-004, IRIDIA, Université Libre de Bruxelles, Belgium**, p. 1 – 63, 2016. Citado nas páginas 93 e 94.

LÓPEZ-IBÁÑEZ, M.; DUBOIS-LACOSTE, J.; CÁCERES, L. P.; BIRATTARI, M.; STÜTZLE, T. The irace package: Iterated racing for automatic algorithm configuration. **Operations Research Perspectives**, v. 3, p. 43 – 58, 2016. Citado nas páginas 92 e 93.

MANBER, U.; ISRANI, S. Pierce point minimization and optimal torch path determination in flame cutting. **Journal of Manufacturing Systems**, v. 3, n. 1, p. 81 – 89, 1984. Citado nas páginas 24, 40, 41, 42, 47 e 111.

MOREIRA, L. M.; OLIVEIRA, J. F.; GOMES, A. M.; FERREIRA, J. S. Heuristics for a dynamic rural postman problem. **Computers & Operations Research**, v. 34, n. 11, p. 3281 – 3294, 2007. Citado nas páginas 39, 40 e 43.

MUNDIM, L. R.; ANDRETTA, M.; QUEIROZ, T. A. de. A biased random key genetic algorithm for open dimension nesting problems using no-fit raster. **Expert Systems with Applications**, v. 81, p. 358 – 371, 2017. Citado nas páginas 32 e 112.

MUNDIM, L. R.; QUEIROZ, T.; ANDRETTA, M. O brkga aplicado em problemas de corte de itens irregulares em um único recipiente. In: **Matemática aplicada à indústria: problemas e métodos de solução**. [S.l.: s.n.], 2016. p. 111 – 136. Citado na página 32.

OLIVEIRA, J. F.; GOMES, A. M.; FERREIRA, J. S. TOPOS - a new constructive algorithm for nesting problems. **OR Spektrum**, v. 22, n. 2, p. 263 – 284, 2000. Citado na página 31.

OLIVEIRA, J. F. C.; FERREIRA, J. A. S. Algorithms for nesting problems. In: **Applied Simulated Annealing**. [S.l.: s.n.], 1993. v. 396, p. 255 – 273. Citado nas páginas 28 e 31.

OLIVEIRA, L. T.; SILVA, E. F.; TOLEDO, F. M. B.; OLIVEIRA, J. F. Modelo integrado de empacotamento de peças irregulares em faixa e caminho mínimo de corte. **Proceeding Series of the Brazilian Society of Computational and Applied Mathematics**, p. 1 – 7, 2018. Citado na página 55.

POTT, A.; GLAAB, H. Optimization problems in a semi-automatic device for cutting leather. In: **Mathematics - Key Technology for the Future**. [S.l.]: Springer Berlin Heidelberg, 2003. p. 609 – 622. Citado nas páginas 47, 48 e 49.

RODRIGUES, A. M.; FERREIRA, J. S. Cutting path as a rural postman problem: solutions by memetic algorithms. **International Journal of Combinatorial Optimization Problems and Informatics**, v. 3, n. 1, p. 31 – 46, 2011. Citado nas páginas 39, 41 e 43.

RODRIGUES, M. O.; TOLEDO, F. M. B. A clique covering mip model for the irregular strip packing problem. **Computers & Operations Research**, v. 87, p. 221 – 234, 2017. Citado nas páginas 35 e 112.

ROQUE, L.; FONTES, D.; FONTES, F. A multi-objective unit commitment problem combining economic and environmental criteria in a metaheuristic approach. **Energy Procedia**, v. 136, p. 362 – 368, 2017. Citado nas páginas 104 e 105.

ROQUE, L. A. C.; FONTES, D. B. M. M.; FONTES, F. A. C. C. A metaheuristic approach to the multi-objective unit commitment problem combining economic and environmental criteria. **Energies**, v. 10, n. 12, p. 1 – 25, 2017. Citado nas páginas 104 e 105.

SEGENREICH, S. A.; BRAGA, L. M. P. F. Optimal nesting of general plane figures: A monte carlo heuristical approach. **Computers & Graphics**, v. 10, n. 3, p. 229 – 237, 1986. Citado na página 31.

SHERIF, S. U.; JAWAHAR, N.; BALAMURALI, M. Sequential optimization approach for nesting and cutting sequence in laser cutting. **Journal of Manufacturing Systems**, v. 33, n. 4, p. 624 – 638, 2014. Citado nas páginas 15, 24, 47, 48, 49 e 111.

SILVA, E. F. **Modelos matemáticos para um problema de caminho de corte**. Dissertação (Mestrado) — Universidade de São Paulo, 2016. Citado nas páginas 24, 101, 103 e 112.

STOYAN, Y.; SCHEITHAUER, G.; GIL, N.; ROMANOVA, T. ϕ-functions for complex 2D-objects. **4OR**, v. 2, p. 69 – 84, 2004. Citado na página 29.

STOYAN, Y.; TERNO, J.; SCHEITHAUER, G.; GIL, N.; ROMANOVA, T. ϕ-functions for primary 2D-objects. **Studia Informatica Universalis**, v. 2, p. 1 – 32, 2002. Citado na página 29.

TAKAHARA, S.; KUSUMOTO, Y.; MIYAMOTO, S. Solution for textile nesting problems using adaptive meta-heuristics and grouping. **Soft Computing**, v. 7, n. 3, p. 154 – 159, 2003. Citado na página 31.

TANGPATTANAKUL, P.; JOZEFOWIEZ, N.; LOPEZ, P. Biased random key genetic algorithm with hybrid decoding for multi-objective optimization. In: **2013 Federated Conference on Computer Science and Information Systems**. [S.l.: s.n.], 2013. p. 393 – 400. Citado nas páginas 104 e 105.

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**, v. 145, n. 2, p. 478 – 487, 2013. Citado nas páginas 32, 33, 34, 35, 58 e 61.

TOSO, R. F.; RESENDE, M. G. **A C++ application programming interface for biased random-key genetic algorithms**. [S.l.], 2012. Citado nas páginas 95 e 105.

USBERTI, F. L.; FRANÇA, P. M.; FRANÇA, A. L. M. The open capacitated arc routing problem. **Computers & Operations Research**, v. 38, n. 11, p. 1543 – 1555, 2011. Citado na página 45.

WÄSCHER, G.; HAUßNER, H.; SCHUMANN, H. An improved typology of cutting and packing problems. **European Journal of Operational Research**, v. 183, n. 3, p. 1109 – 1130, 2007. Citado na página 23.

APÊNDICE

A

ARQUIVOS DO IRACE

Nesse apêndice encontram-se anexados os arquivos utilizados no *irace* para o ajuste dos parâmetros da matheurística proposta no Capítulo 7, sendo eles: o código do *parameters.txt*, o *targetRunner*, o *scenario.txt* e a saída do *irace*, respectivamente.

Parameters

```
# name          switch            type values                    [conditions (using R syntax)]
multipPop        ""           i    (2, 10)
pe               ""           r    (0.1, 0.25)
pm               ""           r    (0.05, 0.3)
rhoe             ""           r    (0.55, 0.95)
GerMax           ""           c    (100, 200, 300, 400, 500, 600, 700, 800, 900, 1000)
```

Target-runner

```
#!/bin/bash
################################################################################
# This script is to tune the nesting software.
#
# PARAMETERS:
# $1 is the ID of the candidate to be evaluated
# $2 is the instance ID
# $3 is the seed
# $4 is the instance name
# The rest ($* after `shift 4') are parameters for running ACOTSP
#
# RETURN VALUE:
# This script should print a single numerical value (the value to be minimized).
################################################################################

# Path to the ACOTSP software:
EXE=$HOME/R/x86_64-pc-linux-gnu-library/3.2/irace/tuning/brkgaAfo-IRACE/main

CONFIG_ID="$1"
INSTANCE_ID="$2"
SEED="$3"
INSTANCE="$4"
# All other parameters are the candidate parameters to be passed to program
shift 4 || error "Not enough parameters to $0"
CONFIG_PARAMS=$*

STDOUT=${INSTANCE}-c${CONFIG_ID}-${INSTANCE_ID}.stdout
STDERR=${INSTANCE}-c${CONFIG_ID}-${INSTANCE_ID}.stderr

# In case of error, we print the current time:
error() {
    echo "`TZ=UTC date`: error: $@" >&2
    exit 1
}

if [ ! -x "${EXE}" ]; then
    error "${EXE}: not found or not executable (pwd: $(pwd))"
fi

# Now we can call ACOTSP by building a command line with all parameters for it
$EXE $INSTANCE ${CONFIG_ID} ${INSTANCE_ID} $SEED ${CONFIG_PARAMS} 1> $STDOUT 2> $STDERR

# The output of the candidate $CONFIG_ID should be written in the file
# c${CONFIG_ID}.stdout (see target runner for ACOTSP).
# Does this file exist?
if [ ! -s "${STDOUT}" ]; then
    # In this case, the file does not exist. Let's exit with a value
    # different from 0. In this case irace will stop with an error.
    error "${STDOUT}: No such file or directory"
fi

# Ok, the file exist. It contains the whole output written by ACOTSP.
# This script should return a single numerical value, the best objective
# value found by this run of ACOTSP. The following line is to extract
# this value from the file containing ACOTSP output.
COST=$(cat ${STDOUT} | grep -o -E 'Best [-+0-9.e]+' | cut -d ' ' -f2)
if ! [[ "$COST" =~ ^[-+0-9.e]+$ ]] ; then
    error "${STDOUT}: Output is not a number"
fi

# Print it!
echo "$COST"

# We are done with our duty. Clean files and exit with 0 (no error).
#rm -f "${STDOUT}" "${STDERR}"
#rm -f best.* stat.* cmp.*
exit 0
```

Scenario

```
##################################################### -*- mode: r -*- #####
## Scenario setup for Iterated Race (iRace).
###########################################################################
## To use the default value of a parameter of iRace, simply do not set
## the parameter (comment it out in this file, and do not give any
## value on the command line).

## File that contains the description of the parameters.
parameterFile = "./parameters.txt"

## Directory where the programs will be run.
execDir = "./Afo-arena"

## File to save tuning results as an R dataset, either absolute path
## or relative to execDir.
logFile = "./iraceNesting.Rdata"

## Directory where tuning instances are located, either absolute path or
## relative to current directory.
trainInstancesDir = ""

## File with a list of instances and (optionally) parameters.
## If empty or NULL, do not use a file.
trainInstancesFile = "./instances-list.txt"

## The script called for each configuration that launches the program to be
## tuned.  See templates/target-runner.tmpl
targetRunner = "../tuning/target-runner"

## The maximum number of runs (invocations of targetRunner) that will
## performed. It determines the (maximum) budget of experiments for the tuning.
maxExperiments = 1000


## Directory where testing instances are located, either absolute or relative
## to current directory.
# testInstancesDir = ""

## File containing a list of test instances and optionally additional
## parameters for them.  If empty or NULL, do not use a file.
# testInstancesFile = ""


## Indicates the number of decimal places to be considered for the
## real parameters.
digits = 2

## A value of 0 silences all debug messages. Higher values provide
## more verbose debug messages.
# debugLevel = 0


## END of scenario file
###########################################################################
```

OUTPUT

```
*****************************************************************************
* irace: An implementation in R of Iterated Race
* Version: 2.4.1844
* Copyright (C) 2010-2017
* Manuel Lopez-Ibanez     <manuel.lopez-ibanez@manchester.ac.uk>
* Jeremie Dubois-Lacoste
* Leslie Perez Caceres    <leslie.perez.caceres@ulb.ac.be>
*
* This is free software, and you are welcome to redistribute it under certain
* conditions.  See the GNU General Public License for details. There is NO
* WARRANTY; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
*
* irace builds upon previous code from the race package:
*
* race: Racing methods for the selection of the best
* Copyright (C) 2003 Mauro Birattari
*****************************************************************************
# installed at: /home/larissa/R/x86_64-pc-linux-gnu-library/3.2/irace
# called with: --scenario ../tuning/scenario.txt
# 2018-09-06 12:41:44 -03: Initialization
# Elitist race
# Elitist new instances: 1
# Elitist limit: 2
# nbIterations: 4
# minNbSurvival: 4
# nbParameters: 5
# seed: 809895371
# confidence level: 0.95
# budget: 1000
# mu: 5
# deterministic: FALSE

# 2018-09-06 12:41:44 -03: Iteration 1 of 4
# experimentsUsedSoFar: 0
# remainingBudget: 1000
# currentBudget: 250
# nbConfigurations: 41
  Markers:
     x No test is performed.
     - The test is performed and some configurations are discarded.
     = The test is performed but no configuration is discarded.
     ! The test is performed and configurations could be discarded but elite configurations
are preserved.

+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
| |   Instance|      Alive|       Best|      Mean best| Exp so far|  W time|  rho|KenW|  Qvar|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
|x|          1|         41|          1|    24.00000000|         41|00:36:09|   NA|  NA|    NA|
|x|          2|         41|          1|    16.66500000|         82|00:09:36|-0.08|0.46|1.1124|
|x|          3|         41|          1|    20.67333333|        123|03:27:13|-0.04|0.31|1.0388|
|x|          4|         41|          1|    17.50500000|        164|00:15:18|-0.04|0.22|1.0458|
|=|          5|         41|          5|    15.48400000|        205|05:28:19|+0.00|0.20|1.0134|
|=|          6|         41|          5|    15.15833333|        246|05:04:25|-0.03|0.15|1.0130|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
Best configuration:           5    mean value:     15.15833333
Description of the best configuration:
  .ID. multipPop   pe   pm rhoe GerMax .PARENT.
5    5         7 0.22 0.13 0.94    200       NA

# 2018-09-07 03:42:47 -03: Elite configurations (first number is the configuration ID; listed
from best to worst according to the sum of ranks):
   multipPop   pe   pm rhoe GerMax
5          7 0.22 0.13 0.94    200
7         10 0.15 0.17 0.60    200
14         4 0.19 0.24 0.78    300
10         4 0.20 0.21 0.67    100
# 2018-09-07 03:42:47 -03: Iteration 2 of 4
# experimentsUsedSoFar: 246
# remainingBudget: 754
# currentBudget: 251
# nbConfigurations: 39
  Markers:
     x No test is performed.
     - The test is performed and some configurations are discarded.
     = The test is performed but no configuration is discarded.
     ! The test is performed and configurations could be discarded but elite configurations
are preserved.
```

```
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
| |   Instance|      Alive|       Best|      Mean best| Exp so far|  W time|  rho|KenW|  Qvar|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
|x|          7|         39|         14|    23.00000000|         39|00:10:37|   NA|  NA|    NA|
|x|          4|         39|         14|    15.50000000|         74|00:15:57|+0.04|0.52|0.9287|
|x|          2|         39|         14|    13.44333333|        109|00:10:06|-0.00|0.33|0.9878|
|x|          5|         39|         14|    11.93250000|        144|04:36:28|+0.01|0.26|0.9843|
|=|          3|         39|         14|    15.28400000|        179|03:05:19|-0.01|0.19|0.9913|
|=|          6|         39|         14|    14.99166667|        214|04:36:42|-0.02|0.15|1.0116|
|=|          1|         39|         14|    16.27857143|        249|00:38:28|-0.02|0.13|0.8618|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
Best configuration:          14    mean value:     16.27857143
Description of the best configuration:
   .ID. multipPop   pe   pm rhoe GerMax .PARENT.
14   14         4 0.19 0.24 0.78    300       NA

# 2018-09-07 17:16:28 -03: Elite configurations (first number is the configuration ID; listed
from best to worst according to the sum of ranks):
   multipPop   pe   pm rhoe GerMax
14         4 0.19 0.24 0.78    300
63         5 0.21 0.12 0.81    900
5          7 0.22 0.13 0.94    200
7         10 0.15 0.17 0.60    200
# 2018-09-07 17:16:28 -03: Iteration 3 of 4
# experimentsUsedSoFar: 495
# remainingBudget: 505
# currentBudget: 252
# nbConfigurations: 35
  Markers:
     x No test is performed.
     - The test is performed and some configurations are discarded.
     = The test is performed but no configuration is discarded.
     ! The test is performed and configurations could be discarded but elite configurations
are preserved.

+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
| |   Instance|      Alive|       Best|      Mean best| Exp so far|  W time|  rho|KenW|  Qvar|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
|x|          8|         35|         14|    25.00000000|         35|00:27:45|   NA|  NA|    NA|
|x|          7|         35|         14|    24.00000000|         66|00:06:17|+0.35|0.67|0.5714|
|x|          1|         35|         14|    24.00000000|         97|00:23:54|+0.18|0.45|0.7692|
|x|          6|         35|         14|    21.38250000|        128|03:36:20|+0.15|0.37|0.7695|
|-|          2|         17|         14|    18.97200000|        159|00:06:22|+0.00|0.20|0.7640|
|=|          5|         17|         14|    17.04333333|        172|01:33:52|+0.04|0.20|0.7927|
|=|          3|         17|         14|    18.70714286|        185|00:59:43|-0.01|0.14|0.8461|
|=|          4|         17|         14|    17.36875000|        198|00:04:03|-0.01|0.12|0.8636|
|=|          9|         17|         14|    18.32777778|        215|00:12:58|-0.03|0.08|0.8937|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
Best configuration:          14    mean value:     18.32777778
Description of the best configuration:
   .ID. multipPop   pe   pm rhoe GerMax .PARENT.
14   14         4 0.19 0.24 0.78    300       NA

# 2018-09-08 00:47:47 -03: Elite configurations (first number is the configuration ID; listed
from best to worst according to the sum of ranks):
    multipPop   pe   pm rhoe GerMax
14          4 0.19 0.24 0.78    300
63          5 0.21 0.12 0.81    900
96          5 0.19 0.22 0.81    300
104         6 0.23 0.09 0.89    900
# 2018-09-08 00:47:47 -03: Iteration 4 of 4
# experimentsUsedSoFar: 710
# remainingBudget: 290
# currentBudget: 290
# nbConfigurations: 32
  Markers:
     x No test is performed.
     - The test is performed and some configurations are discarded.
     = The test is performed but no configuration is discarded.
     ! The test is performed and configurations could be discarded but elite configurations
are preserved.

+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
| |   Instance|      Alive|       Best|      Mean best| Exp so far|  W time|  rho|KenW|  Qvar|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
|x|         10|         32|         14|    24.00000000|         32|00:21:38|   NA|  NA|    NA|
|x|          7|         32|         14|    23.50000000|         60|00:04:50|+0.00|0.50|0.4844|
|x|          9|         32|        128|    24.00000000|         88|00:19:32|-0.09|0.28|0.6679|
```

```
|x|          3|         32|        128|    25.17250000|        116|02:11:08|+0.05|0.28|0.6786|
|=|          1|         32|        128|    24.93800000|        144|00:18:25|+0.03|0.23|0.5525|
|=|          6|         32|        128|    23.03666667|        172|03:28:07|+0.07|0.23|0.5956|
|=|          2|         32|        128|    21.07857143|        200|00:05:00|+0.06|0.19|0.6637|
|=|          5|         32|        128|    19.36875000|        228|03:30:55|+0.05|0.17|0.6875|
|=|          8|         32|         14|    20.10555556|        256|00:20:11|+0.04|0.15|0.7168|
|=|          4|         32|         14|    18.89500000|        284|00:07:43|+0.04|0.13|0.7334|
+-+-----------+-----------+-----------+---------------+-----------+--------+-----+----+------+
Best configuration:          14    mean value:     18.89500000
Description of the best configuration:
   .ID. multipPop   pe   pm rhoe GerMax .PARENT.
14   14         4 0.19 0.24 0.78    300       NA

# 2018-09-08 11:35:21 -03: Elite configurations (first number is the configuration ID; listed
from best to worst according to the sum of ranks):
    multipPop   pe   pm rhoe GerMax
14          4 0.19 0.24 0.78    300
63          5 0.21 0.12 0.81    900
128         3 0.19 0.25 0.80    300
130         5 0.20 0.24 0.82    300
# 2018-09-08 11:35:21 -03: Stopped because there is not enough budget left to race more than
the minimum (4)
# You may either increase the budget or set 'minNbSurvival' to a lower value
# Iteration: 5
# nbIterations: 5
# experimentsUsedSoFar: 994
# timeUsed: 0
# remainingBudget: 6
# currentBudget: 6
# number of elites: 4
# nbConfigurations: 4
# Best configurations (first number is the configuration ID; listed from best to worst
according to the sum of ranks):
    multipPop   pe   pm rhoe GerMax
14          4 0.19 0.24 0.78    300
63          5 0.21 0.12 0.81    900
128         3 0.19 0.25 0.80    300
130         5 0.20 0.24 0.82    300
# Best configurations as commandlines (first number is the configuration ID; same order as
above):
14   4 0.19 0.24 0.78 300
63   5 0.21 0.12 0.81 900
128  3 0.19 0.25 0.8 300
130  5 0.2 0.24 0.82 300
```

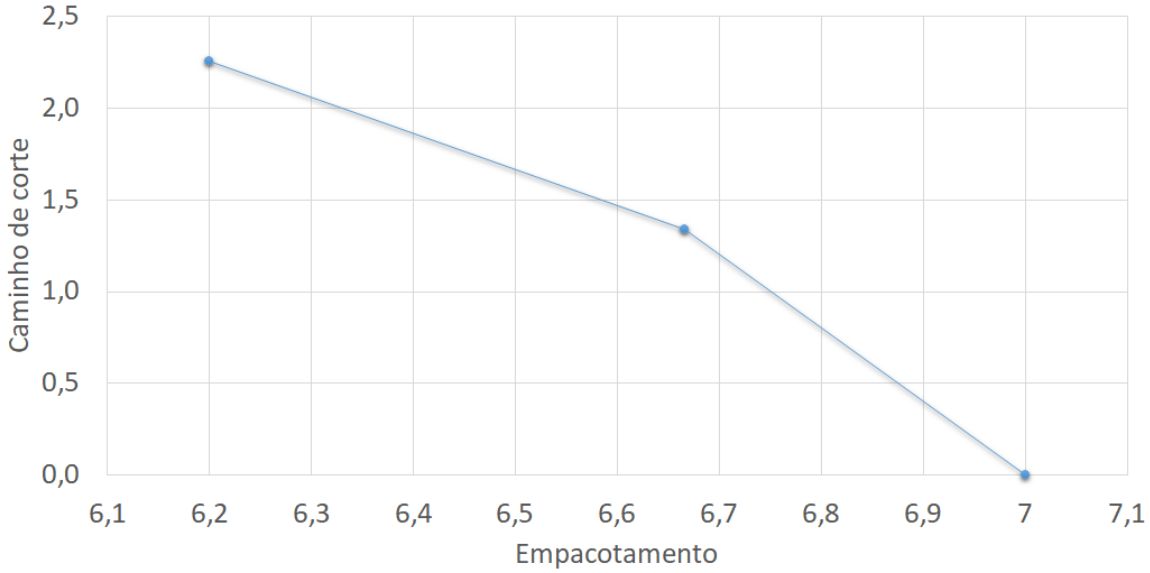
APÊNDICE

B

RESULTADOSS OBTIDOS PELO BRKGAINTEGRADO

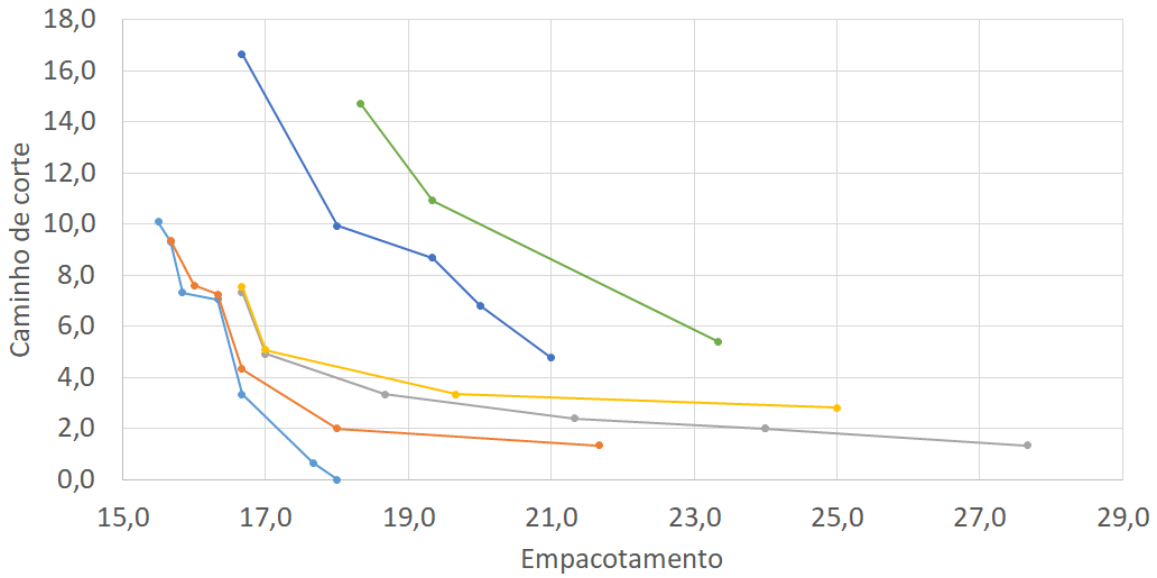
Nesse apêndice são apresentados os gráficos com as soluções obtidas pelo BRKGAIntegrado e as fronteira para as instâncias apresentadas no Capítulo 8.

Figura 40 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância three.



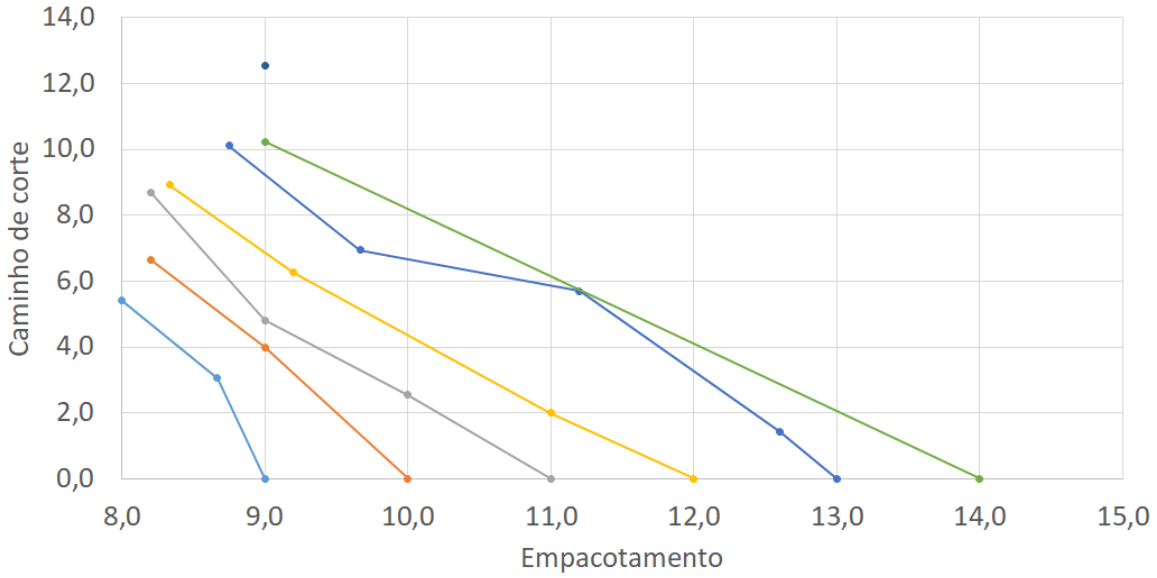
Fonte: Elaborada pela autora.

Figura 41 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância three3.



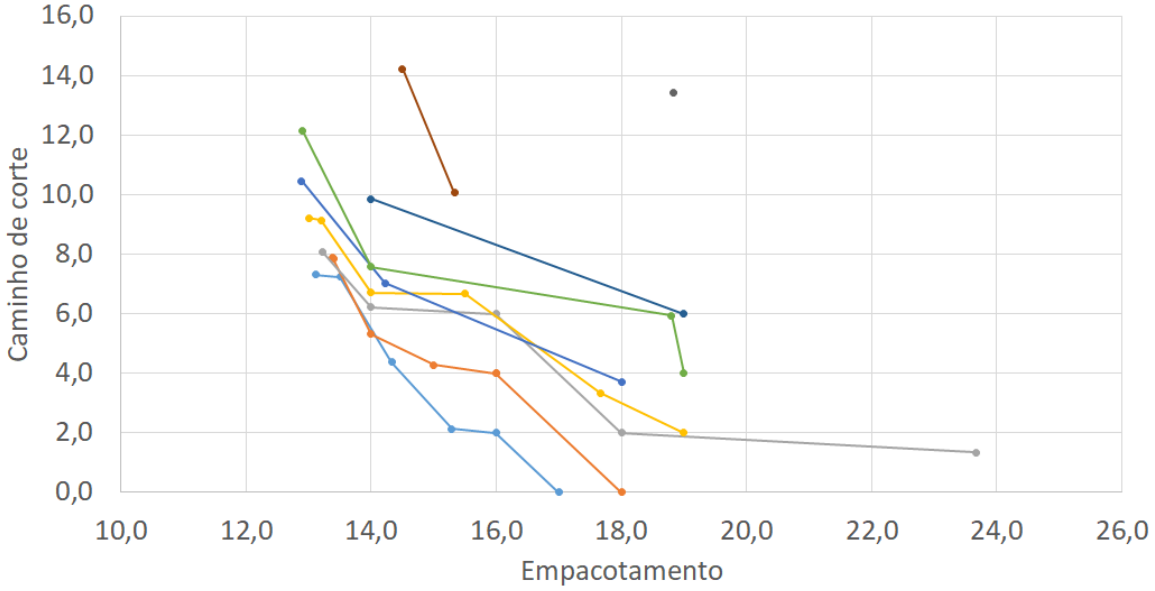
Fonte: Elaborada pela autora.

Figura 42 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância three2w9.



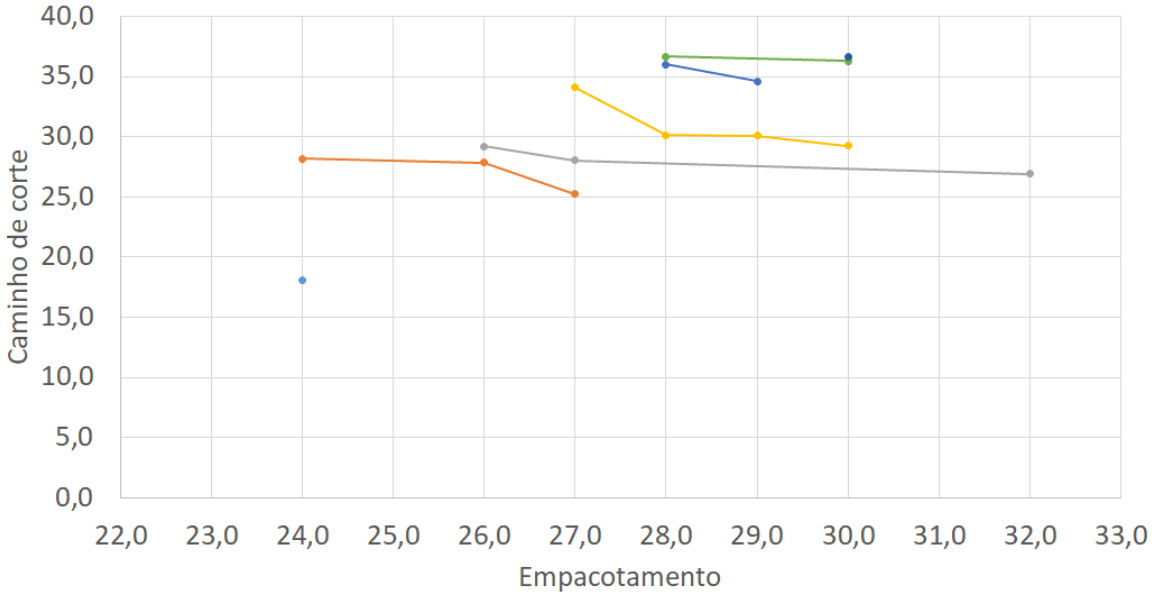
Fonte: Elaborada pela autora.

Figura 43 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância three3w9.



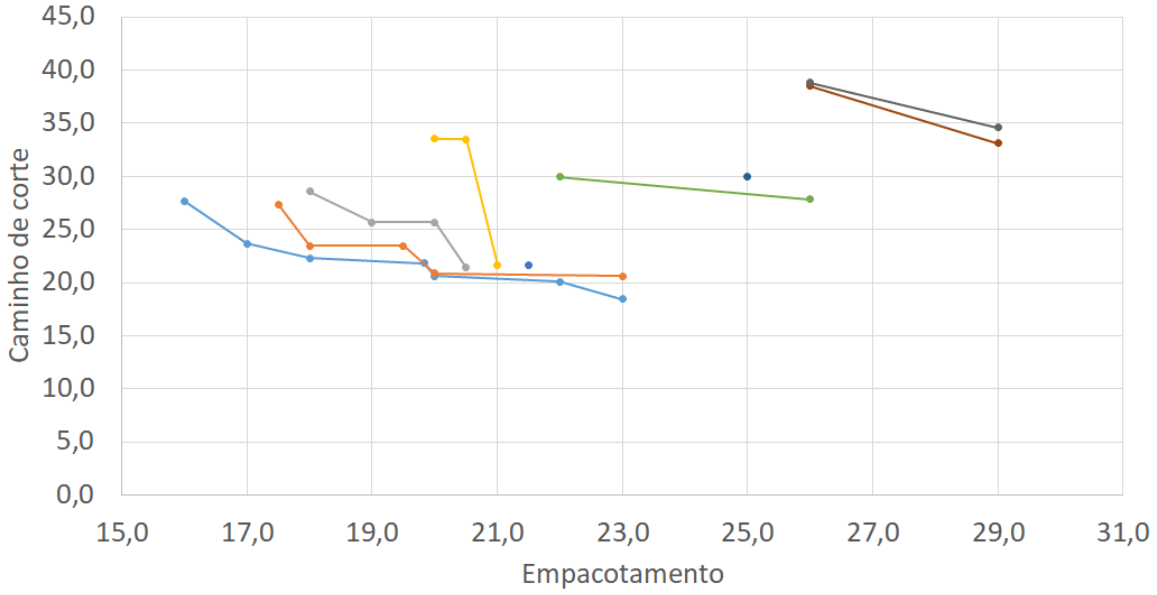
Fonte: Elaborada pela autora.

Figura 44 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância shapes4.



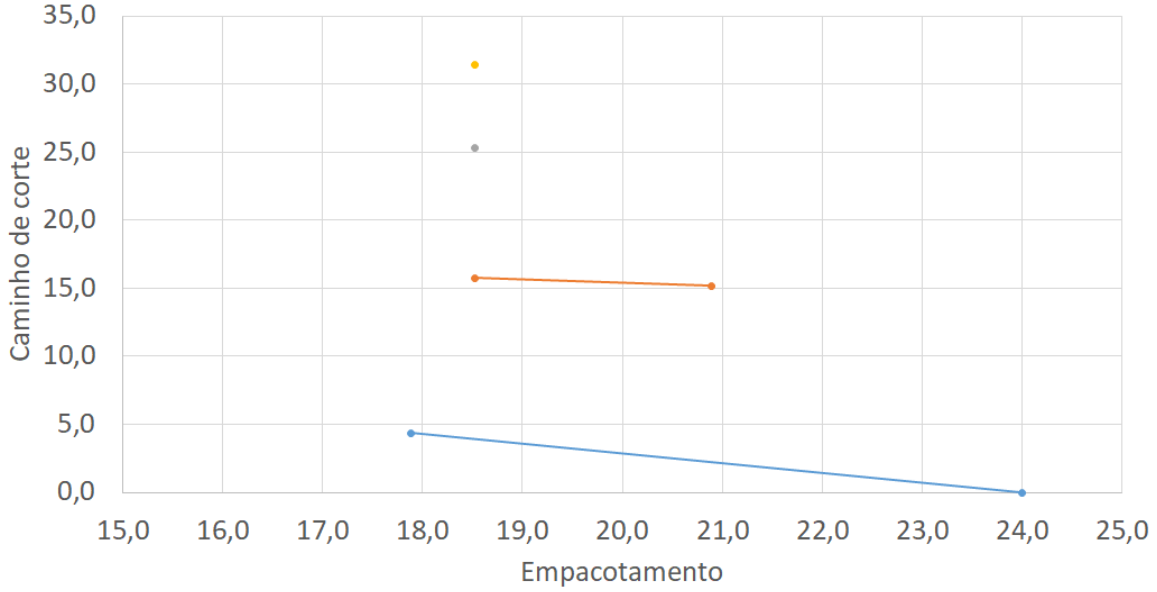
Fonte: Elaborada pela autora.

Figura 45 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância shapes4N.



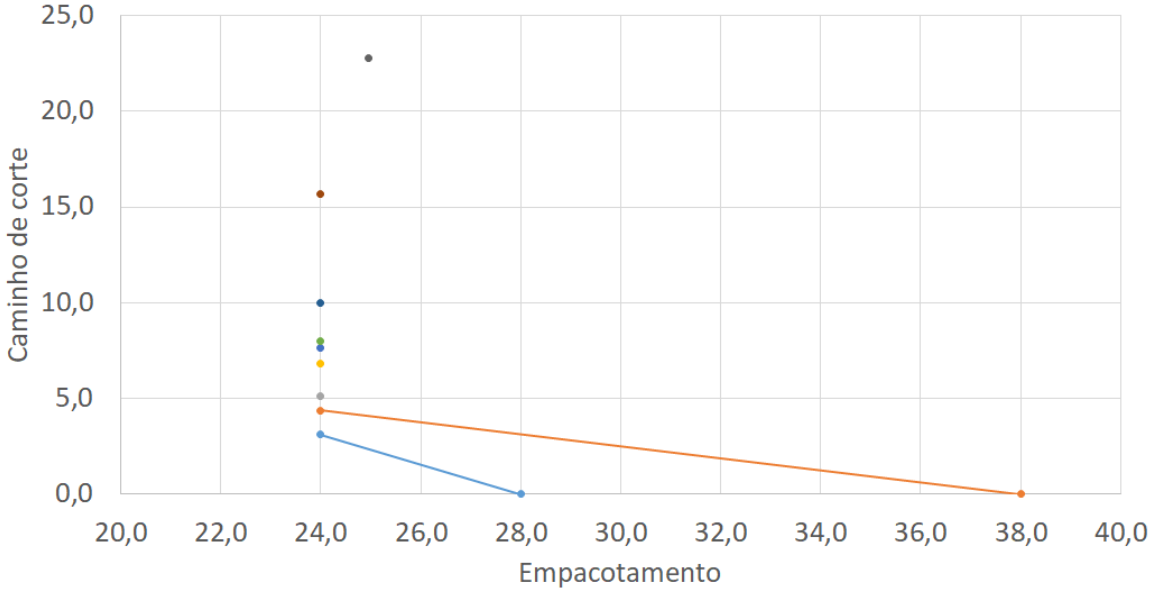
Fonte: Elaborada pela autora.

Figura 46 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância fu5.



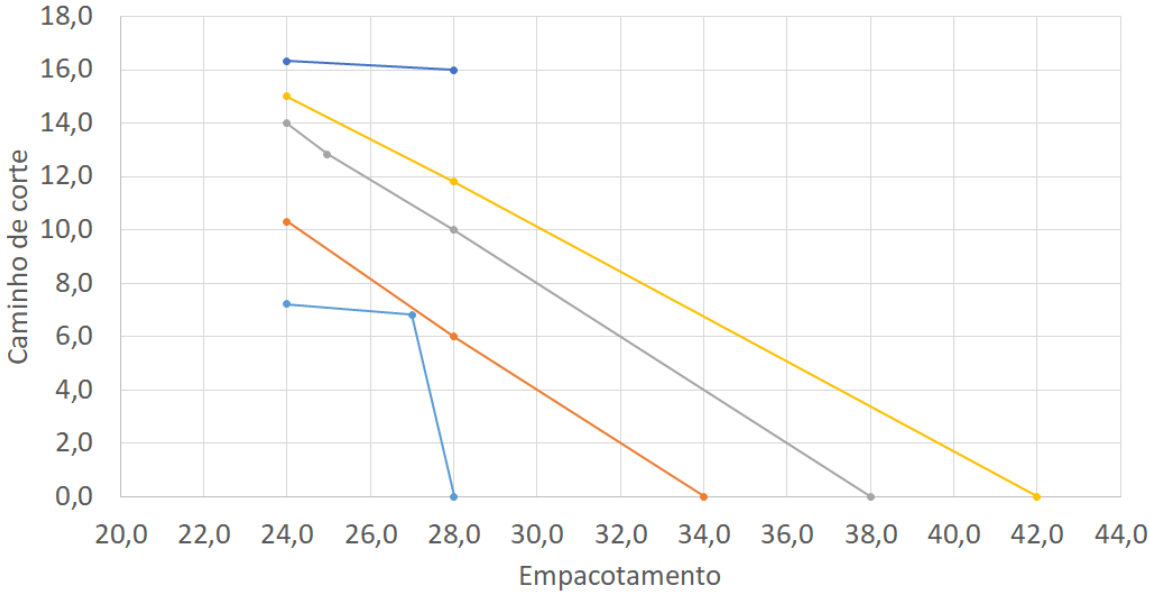
Fonte: Elaborada pela autora.

Figura 47 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância fu6.



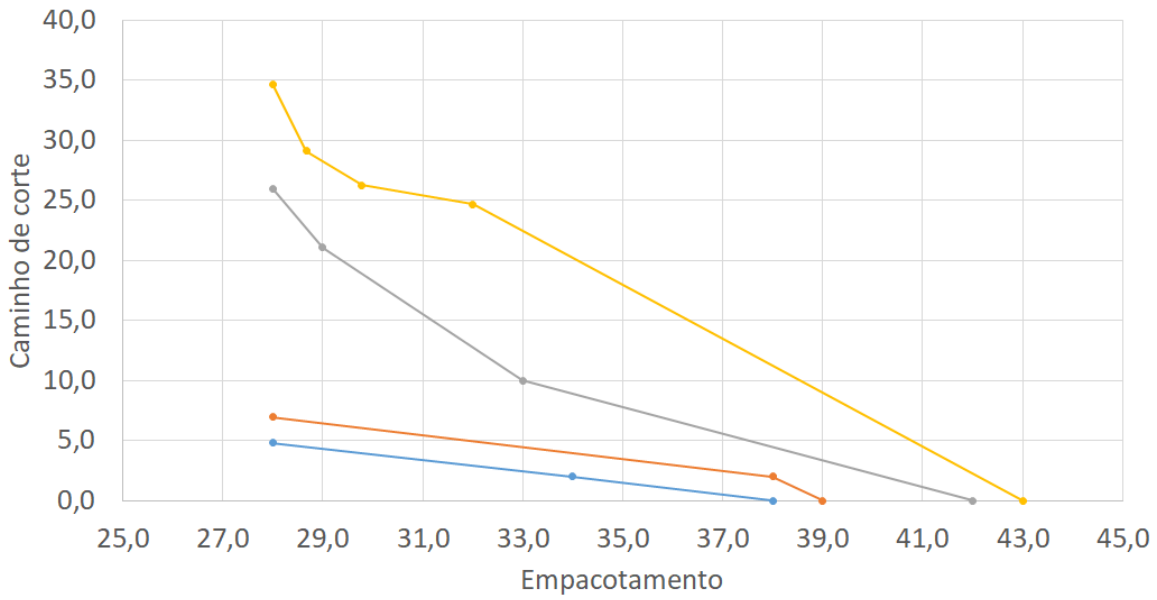
Fonte: Elaborada pela autora.

Figura 48 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância fu7.



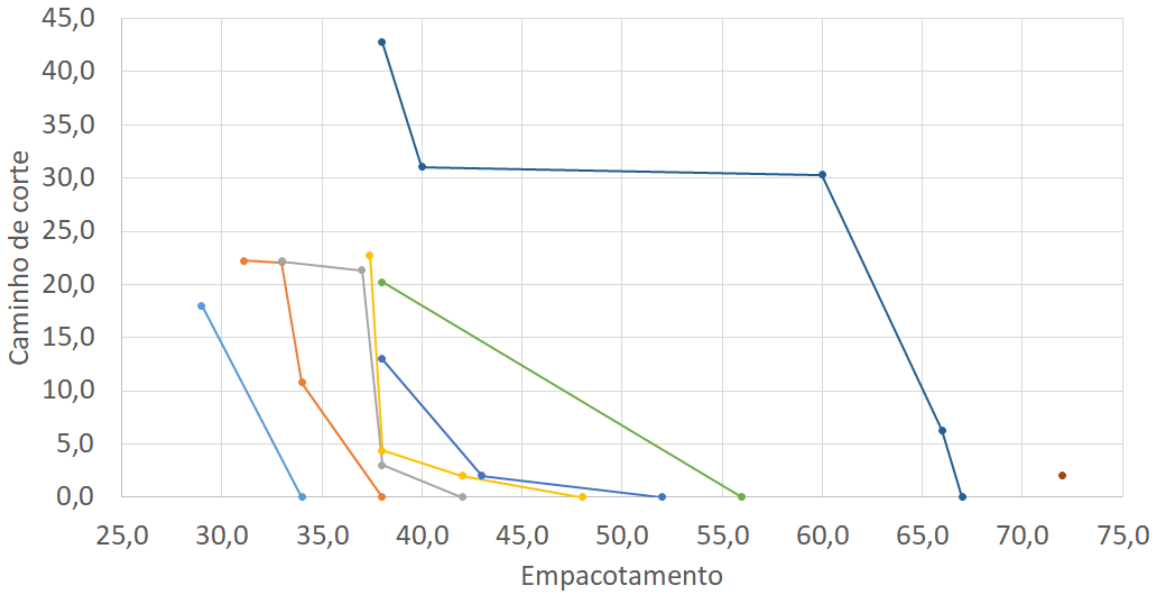
Fonte: Elaborada pela autora.

Figura 49 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância fu8.



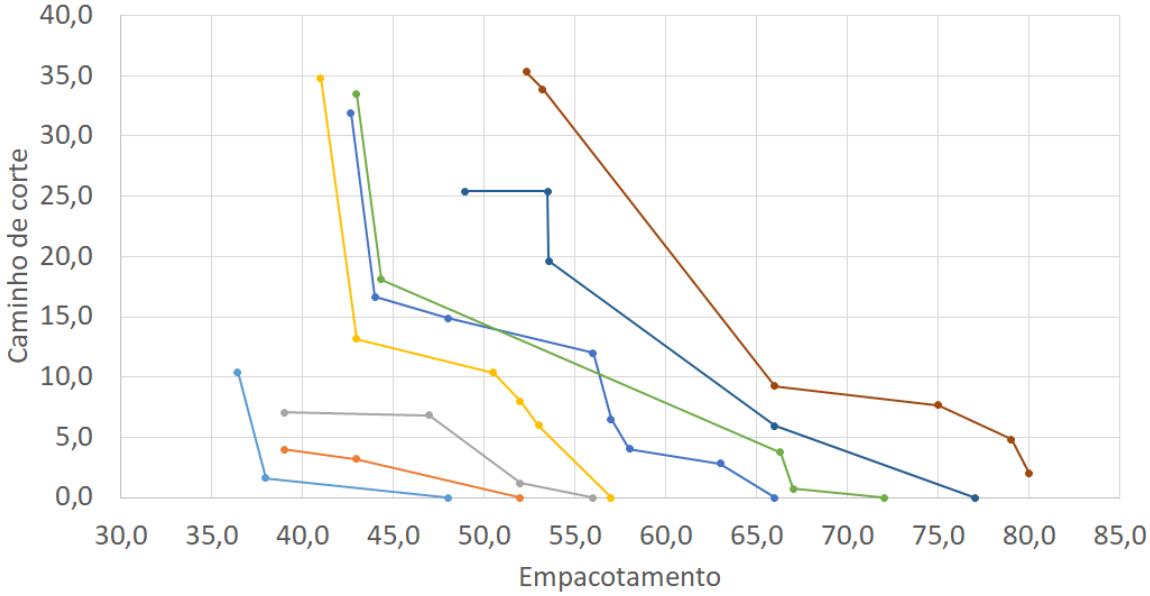
Fonte: Elaborada pela autora.

Figura 50 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância fu9.



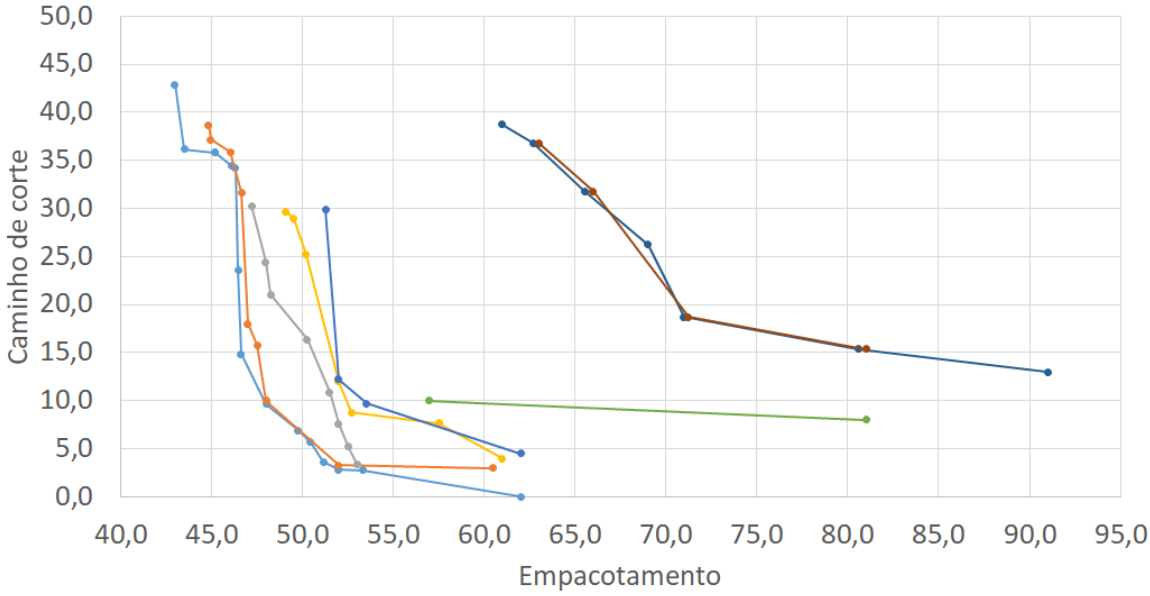
Fonte: Elaborada pela autora.

Figura 51 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância fu10.



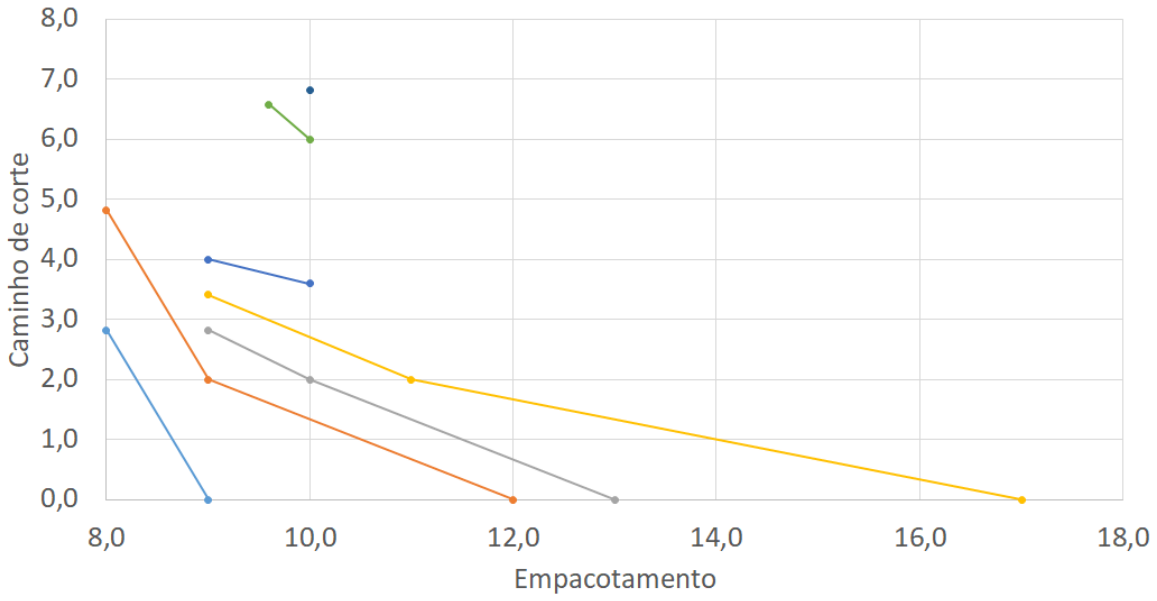
Fonte: Elaborada pela autora.

Figura 52 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância fu12.



Fonte: Elaborada pela autora.

Figura 53 – Resultados do BRKGAIntegrado obtidos em 10 rodadas para a instância rco1.



Fonte: Elaborada pela autora.

ICMC USP
SÃO CARLOS