Universidade do Minho
Escola de Engenharia

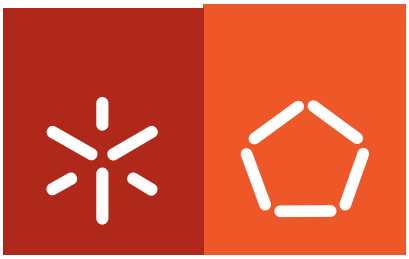
Elsa Marília da Costa Silva

Modelos e métodos de otimização para problemas de corte e empacotamento a duas dimensões

Novembro de 2011

Modelos e métodos de otimização para problemas de corte e empacotamento a duas dimensões
Elsa Marília da Costa Silva
UMinho | 2011


FCT Fundação para a Ciência e a Tecnologia



Universidade do Minho
Escola de Engenharia

Elsa Marília da Costa Silva

Modelos e métodos de otimização para problemas de corte e empacotamento a duas dimensões

Programa Doutoral em Engenharia Industrial e de Sistemas

Trabalho efectuado sob a orientação do
Professor Doutor Filipe Pereira Pinto da Cunha e Alvelos
e do
Professor Doutor José Manuel Vasconcelos Valério de Carvalho

Novembro de 2011

Agradecimentos

O caminho faz-se caminhando, e o caminho percorrido até à consecução desta tese foi acompanhado por um somatório de pequenas e grandes realizações pessoais e profissionais. A caminhada não foi solitária, e deste modo, gostaria de expressar o meu sincero agradecimento a todos quantos ao longo destes anos me acompanharam.

As minhas primeiras palavras de agradecimento são, como de outra forma não poderia ser, para os meus orientadores. A ambos agradeço a partilha de conhecimentos e experiências, assim como a confiança que depositaram em mim para a realização deste projeto de investigação.

Ao Professor Filipe Alvelos, agradeço todos os bons ensinamentos que me foi transmitindo desde a licenciatura e por me fazer despertar o gosto pela investigação operacional. O seu estímulo e incentivo foram essenciais para a concretização deste trabalho, agradeço a sua dedicação assim como a sua disponibilidade e as oportunidades concedidas.

Ao Professor Valério de Carvalho, expresso o meu reconhecimento pelos seus preciosos conselhos e disponibilidade. Os seus profundos conhecimentos de otimização combinatória e programação inteira são uma fonte de inspiração.

É de toda a justiça afirmar que os amigos do laboratório foram um contributo importante ao longo destes anos de investigação, pelos momentos agradáveis de descontração, ânimo e amizade, de entre eles destacam-se: o Pedro, o Tiago, a Carina, o Tó e o Telmo. Uma palavra especial de apreço é dirigida à Rita, que esteve sempre presente desde os primeiros passos no caminho do doutoramento, juntas enfrentamos novos desafios e assistimos a um crescimento mútuo, cada uma percorrendo o seu próprio caminho. Por todos os momentos que passamos juntas, posso afirmar com certeza que a nossa amizade permanecerá no tempo.

Ao Eng. Acácio, agradeço todo o seu profissionalismo e valioso suporte técnico concedido ao longo destes anos, à Conceição agradeço todo o trabalho administrativo.

Agradeço também aos Professores Marcos Arenales, Franklina Toledo, Maristela Santos e Vitória Pureza pela hospitalidade e a partilha de conhecimentos durante a minha estadia de três meses no Brasil. Ao LOT (Laboratório de Otimização) agradeço a amabilidade com que me receberam e integraram, em especial à Tamara, ao Marcos e ao Márcio.

Além disso, agradeço à FCT (Fundação para a Ciência e Tecnologia) pelo apoio financeiro concedido para a realização desta tese de doutoramento, através da bolsa SFRH/BD/42259/2007. Agradeço também à Universidade do Minho, em particular ao Departamento de Produção e Sistemas da Escola de Engenharia, pelo apoio técnico, equipamentos e condições de trabalho para o desenvolvimento da minha investigação.

Aos meus amigos mais próximos agradeço o apoio e as palavras de incentivo, em especial à Francisca pelas sugestões de revisão na componente de escrita da tese.

Ao Filipe, expresso a minha gratidão pelas constantes palavras de encorajamento, a compreensão nos momentos de ausência e a inesgotável paciência, a sua presença foi um contributo fundamental para a execução deste projeto.

Por último, e não menos importante, à minha família, em particular aos meus pais e ao meu irmão, agradeço todo o amor e apoio permanente e incondicional.

Braga, Novembro de 2011

Elsa Silva

Resumo

Nesta tese, propõem-se modelos e métodos de otimização para problemas de corte e empacotamento a duas dimensões.

Os problemas de corte e empacotamento são problemas com os quais muitas indústrias têm de lidar no decorrer do seu processo produtivo.

De um modo genérico estes problemas consistem na divisão de um recurso, materiais sólidos no caso do corte e espaço utilizável no caso do empacotamento, em partes mais pequenas otimizando um determinado objetivo. Diferentes variantes e objetivos podem ser considerados dependendo das aplicações onde o problema tem origem. Devido à sua vasta aplicabilidade estes problemas têm sido muito estudados nas últimas décadas.

Nesta tese, são propostos novos modelos de programação inteira com tamanho pseudo-polinomial para diferentes variantes do problema de corte bidimensional e para o problema integrado de corte bidimensional e dimensionamento de lotes. É também proposta uma aplicação de um algoritmo SearchCol (pesquisa meta-heurística por geração de colunas), que combina geração de colunas com meta-heurísticas ao problema de empacotamento bidimensional.

O primeiro problema de corte estudado tem restrições de guilhotina e limitação no número de cortes (estágios) possíveis. Para este problema apresenta-se um modelo de programação inteira cujo tamanho, como se prova, é pseudo-polinomial. Neste modelo cada variável de decisão está associada com o corte de um item de uma placa ou de parte de uma placa, que resulte de cortes anteriores (placas residuais).

Além das restrições de guilhotina e do número de estágios foram estudadas outras variantes do problema. O modelo de programação inteira proposto foi

estendido para o problema de corte bidimensional também estagiado em que se permite a rotação dos itens, para o problema de corte em que os itens podem ser cortados de placas com tamanhos diferentes, para o problema de minimizar o tempo despendido no corte dos itens das placas, para o problema de minimizar os desperdícios e ainda para o problema de corte bidimensional onde se atribuem valores às placas residuais de sobra.

O estudo dos problemas de corte não pode ser analisado de uma forma isolada no processo produtivo, além disso resolver um problema de corte em cada período de planeamento, pode resultar em soluções de fraca qualidade, isto é, com desperdício excessivo, quando considerado um conjunto de períodos de planeamento. Com o objetivo de abordar esta característica é estudado também o problema integrado de corte bidimensional e dimensionamento de lotes. Para este problema são apresentados dois modelos de programação inteira.

O problema de empacotamento bidimensional foi também estudado e para a sua resolução foi utilizado um algoritmo SearchCol, que tem por base um novo modelo de decomposição para o problema de empacotamento bidimensional.

Testes computacionais foram realizadas para todos os modelos de programação inteira propostos e para a aplicação do SearchCol ao problema de empacotamento bidimensional, para tal foram utilizadas instâncias reais provenientes da indústria do mobiliário e instâncias da literatura.

De um modo geral, os resultados computacionais comprovam a eficiência dos modelos e métodos propostos.

Abstract

In this thesis, optimization models and methods are proposed for the two dimensional cutting stock and bin packing problems.

The cutting and packing problems are problems that many industries have to deal during the production process.

Generally these problems consist in a division of a resource in small parts, solid materials in the case of cutting and usable space in the case of packaging, in order to optimize a certain objective. Different variants and objectives may be considered depending on the applications where the problem appeared. Due to the wide applicability of these problems, they have been very studied in recent decades.

In this thesis, new models of integer programming pseudo-polynomial in size are proposed for different variants of the two-dimensional cutting stock problem and for the integrated two-dimensional cutting stock and lot sizing problems. It is also proposed an application of an algorithm SearchCol (metaheuristic search by column generation), which combines column generation with meta-heuristics, to the two-dimensional bin packing problem.

Firstly it is studied the two-dimensional cutting stock problem with guillotine constraints and with a limitation in the number of possible cuts (stages). For this problem, it is proposed an integer programming model, which size is proved to be pseudo-polynomial. In this model each decision variable is associated with cutting an item from a plate or from a part of a plate, resulting from previous cuts (residual plate).

Besides the guillotine and the number of stages constraints, other variants of the problem have been studied. The proposed integer programming model was

extended to the two-dimensional cutting stock problem also staged, which allows items rotation, for the cutting stock problem in which the items can be cut form plates with different sizes, for the problem of minimizing the time spent in the cutting process of the items from the plates, for the problem of minimizing the wastage and also for the cutting stock problem in which values are assigned to the residual plates.

The study of the cutting stock problems can not be analysed in isolated way in the production process, also solving a cutting stock problem in each planning period, can result in poor quality solutions, i.e., with excessive waste, when considered a set of planning periods. In order to address this feature, it is also studied the integrated two-dimensional cutting stock and lot-sizing problem. For this problem are proposed two integer programming models.

The two-dimensional bin packing problem has also been studied and for solving it a SearchCol algorithm was used, which is based on a new decomposition model for the two-dimensional bin packing problem.

Computational tests were conducted for all the proposed integer programming models and for the application of the SearchCol to the two-dimensional bin packing problem, real instances from furniture companies and instances from the literature are used.

Generally, the computational results prove the efficiency of the proposed models and methods.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução geral

Neste capítulo faz-se uma introdução geral ao trabalho desenvolvido. Apresenta-se o âmbito da tese, em que contexto é que surgem os problemas de corte, a sua relevância económica e as dificuldades inerentes à sua resolução. Detalham-se ainda as principais contribuições e apresenta-se a estrutura da tese.

1.1 Âmbito da tese

Os problemas de corte e empacotamento, de um modo geral, consistem na divisão de um recurso em partes mais pequenas, de acordo com uma procura e um tamanho pré-estabelecidos, otimizando uma determinada função objetivo, por exemplo, a minimização dos desperdícios, a minimização dos custos de produção ou a maximização do lucro.

Estes problemas estão presentes em diferentes processos industriais tais como, o corte de bobinas de papel, de alumínio ou de barras de aço, de placas de metal, de madeira ou de vidro, de rolos de tecido, de pele, da divisão do espaço disponível num contentor ou numa palete ou da disposição de contentores em navios, entre outros. Os diferentes ambientes industriais em que surgem estes problemas têm as suas especificidades, conduzindo a diferentes exigências práticas e objetivos.

Nestas indústrias, reduções substanciais dos desperdícios significam uma melhoria da eficiência, conduzindo a um aumento da competitividade e do lucro, o que, de certo modo, motiva a investigação académica em modelos de otimização para os problemas de corte e empacotamento.

Devido à sua vasta aplicabilidade industrial, os problemas de corte e empacotamento revestem-se de uma grande importância, no entanto, a sua aparente simplicidade de definição é contrastada pela sua complexidade computacional, uma vez que se tratam de problemas NP-difíceis (Garey & Johnson 1990), tornando-os em problemas de otimização combinatória muito interessantes.

Diferentes variantes do problema podem ser consideradas, normalmente associadas a diferentes indústrias onde o problema surge, e podem existir condicionalismos técnicos relacionados com as máquinas de corte, que podem impor que os cortes sejam guilhotináveis (de um lado ao outro da placa) e ainda que haja um limite no número de estágios (conjuntos de cortes verticais ou horizontais)

possíveis. São exemplos os problemas considerados nesta tese, que têm origem na indústria do mobiliário, e consistem no corte de placas de matéria-prima em partes que serão utilizadas na produção de móveis.

Os problemas de corte estão bastante relacionados com os problemas de empacotamento. Nos problemas de empacotamento existe uma variedade muito heterogénea das partes, isto é, existem muitos tipos de peças com procuras pequenas (no limite, todas as peças são diferentes). Por outro lado, nos problemas de corte existe uma variedade pouco heterogénea das peças, isto é, existe um número pequeno de tipos de peças e cada tipo tem uma procura elevada.

Em termos práticos esta distinção pode ocorrer devido à política de produção de uma determinada empresa, por exemplo se é utilizada uma política de produção orientada para o inventário, as procuras de cada item tomam valores elevados, enquanto que numa produção centralizada na encomenda a procura de cada item tem um valor menor.

1.2 Contribuições

A contribuição desta tese está intrinsecamente relacionada com os métodos de otimização propostos para as diferentes variantes dos problemas de corte e empacotamento a duas dimensões estudados.

É proposto um modelo de programação inteira de tamanho pseudo-polinomial para o problema de corte a duas dimensões que pode ser utilizado para resolver problemas em que o padrão de corte pode ser do tipo dois-estágios exato, dois-estágios não-exato, três estágios exato e três-estágios não-exato. Este modelo é estendido para resolver também outras variantes dos problemas de corte, como os problemas de corte em que é permitida a rotação dos itens, problemas em que os itens podem ser cortados de placas com tamanhos diferentes, problemas em

que o tempo dispendido nos cortes tenham uma grande relevância, assim como problemas de minimização dos desperdícios e ainda problemas em que sejam atribuídos valores a placas residuais de sobra.

No problema de corte a duas dimensões existe um conjunto de itens retangulares de dimensões variadas que se pretende cortar de um conjunto de placas, também retangulares, de acordo com um determinado objetivo, tipicamente, a minimização do desperdício total. Numa organização este problema é repetitivo, tendo em conta que é considerado no início de cada período de planeamento, sendo a solução ótima de problemas de corte procurada em períodos de planeamento da produção consecutivos. Nesta tese estuda-se também o problema de corte estendido de uma encomenda para uma sequência de encomendas, reaproveitando placas residuais de encomendas anteriores e considerando também a possibilidade de antecipar a produção de alguns itens considerando os respetivos custos de armazenamento. Na literatura esta característica nunca foi abordada. Este problema é designado por problema integrado de corte bidimensional e dimensionamento de lotes e para a sua resolução contribuímos com dois modelos de programação inteira, também de tamanho pseudo-polinomial.

Para o problema de empacotamento bidimensional contribuímos com o novo modelo, a relaxação linear é resolvida através de geração de colunas e aplica-se a recente combinação de geração de colunas com meta-heurísticas, SearchCol (pesquisa meta-heurística por geração de colunas) proposta por Alvelos et al. (2010). No algoritmo de geração de colunas proposto as variáveis do problema mestre restrito correspondem a uma seleção de padrões de empacotamento agrupados pelo item que está posicionado no canto inferior esquerdo. Cada subproblema propõe padrões de empacotamento inicializados por um item diferente, fazendo com que o número de subproblemas seja igual ao número de itens.

1.3 Estrutura da tese

Nesta tese são abordados problemas de corte e de empacotamento a duas dimensões e a sua estrutura é descrita em seguida.

O capítulo 2 é dedicado à definição dos problemas de corte e empacotamento abordados na tese e à revisão dos diferentes métodos e modelos utilizados para resolver estes problemas. Os problemas de corte e empacotamento têm origem em diferentes indústrias, o que conduziu a que cada problema possua diferentes especificidades e objetivos. Neste capítulo procede-se à definição dos problemas de corte e empacotamento abordados, tendo por base a indústria do mobiliário onde o problema considerado na tese teve origem.

Devido às diferentes aplicações nas diferentes indústrias, estes problemas surgem na literatura sob diferentes designações, embora tenham em comum a mesma estrutura lógica. Nesse sentido surgiram na literatura duas tipologias, uma em 1990 e outra em 2007, com o objetivo de categorizar todos os problemas de corte e empacotamento assim como a literatura existente. Estas tipologias são apresentadas e os problemas abordados na tese são classificados segundo as mesmas.

Apresenta-se também uma revisão dos trabalhos da literatura que utilizam modelos de geração de colunas para resolver problemas de corte e de empacotamento a duas dimensões, assim como de modelos de afetação para os problemas de empacotamento bidimensional. O capítulo termina com uma revisão das abordagens heurísticas que têm sido utilizadas na literatura para resolver os problemas de corte e de empacotamento abordados.

No capítulo 3 é proposto um modelo de programação inteira para o problema de corte bidimensional. Este modelo foi desenvolvido considerando condicionalismos técnicos relacionados com as máquinas, que limitam o número de cortes a dois e a três estágios. O modelo baseia-se na consideração explícita de como os

itens podem ser obtidos das placas, tem tamanho pseudo-polinomial e pode ser resolvido por um *solver* genérico de programação inteira.

Além disso, é flexível e permite modelar diferentes características que podem ser muito importantes em problemas práticos, como a utilização de placas de diferentes tamanhos, a inclusão de valores para as peças residuais, a minimização do comprimento dos cortes e a rotação dos itens.

São efetuados testes computacionais em instâncias reais provenientes de duas empresas do sector do mobiliário e em instâncias da literatura. O resultados obtidos com o modelo proposto são comparados com os obtidos com dois modelos de programação inteira da literatura e com dois modelos de geração de colunas.

No capítulo 4 é abordado o problema de corte bidimensional combinado com o problema de dimensionamento de lotes.

O planeamento da produção é um processo repetitivo que acontece no início de cada período de planeamento, baseado nas encomendas dos clientes. Desta forma, procurar a solução ótima para um problema de corte num único período de planeamento é diferente de uma estratégia que avalie se se deve antecipar ou não a produção de itens e o armazenamento de placas residuais, para utilização em períodos de planeamento da produção subsequentes, passando-se assim de uma estratégia de procura de um ótimo local, para uma política de procura de um ótimo global. É então neste contexto que surge o problema integrado de corte bidimensional e dimensionamento de lotes.

Neste capítulo são apresentados dois modelos de programação inteira para resolver este problema. No modelo proposto o objetivo é minimizar os custos de produção e os custos de armazenamento de itens e de placas residuais.

São apresentados testes computacionais com instâncias reais e os modelos propostos são comparados com duas heurísticas.

No capítulo 5 é proposta a aplicação do *framework* SearchCol ao problema de empacotamento bidimensional. O SearchCol é um método que combina geração de colunas com meta-heurísticas para resolver problemas de otimização combinatória que sejam decomponíveis.

Para aplicação do SearchCol é necessário um modelo de decomposição. Sendo assim, neste capítulo é proposto um modelo de decomposição e um novo algoritmo de geração de colunas para o problema de empacotamento bidimensional. No modelo de decomposição proposto, cada subproblema está associado a um item, existindo assim tantos subproblemas quanto o número de itens.

As pesquisas meta-heurísticas utilizadas são a pesquisa local multi-início (*multi-start local search*) e a pesquisa em vizinhanças variáveis (*VNS*). Além disso o SearchCol inclui a utilização de perturbações na geração de colunas para promover a melhoria da solução incumbente. Deste modo, são também discutidas estratégias de perturbação.

Testes computacionais foram realizados em instâncias da literatura e as soluções obtidas para as várias variantes da aplicação do SearchCol ao problema de empacotamento bidimensional são comparadas com um modelo de programação inteira da literatura.

Por fim, no capítulo 6 são retiradas as conclusões finais do trabalho desenvolvido.

Capítulo 2

Problemas de corte e empacotamento

Neste capítulo são apresentadas duas classificações dos problemas de corte e empacotamento e uma revisão bibliográfica dos métodos de solução para alguns destes problemas. O capítulo inicia-se com uma classificação dos problemas de corte e empacotamento segundo duas grandes tipologias. Neste contexto, uma classificação dos problemas de corte, abordados na tese, é efetuada. A revisão bibliográfica está dividida em três grandes grupos, o primeiro engloba os métodos de geração de colunas, o segundo os modelos de afetação e o terceiro grupo reúne os métodos heurísticos.

2.1 Introdução

Os problemas de corte e empacotamento têm origem numa multiplicidade de indústrias e de aplicações do mundo real, de entre os quais se destacam o corte de matérias-primas como: o metal, o vidro, o papel, a madeira, a pele, o plástico e o tecido e também o empacotamento de peças em: veículos, paletes, contentores e caixas.

De um modo genérico os problemas de corte e empacotamento consistem na divisão de um recurso, materiais sólidos no caso do corte e espaço utilizável no caso do empacotamento, em partes mais pequenas otimizando um determinado objetivo. O objetivo poderá ser por exemplo a minimização do número de placas de vidro, de madeira ou de outro material necessárias para o corte de um conjunto de peças procuradas ou o empacotamento do maior número possível de caixas num veículo, ou ainda selecionar de entre um conjunto de contentores com diferentes valores associados os que permitam um lucro máximo quando empacotados num navio.

Para a definição de um problema de corte ou de empacotamento são necessários dois conjuntos de dados, o conjunto dos materiais a dividir ou a cortar, que são designados por *objetos* e o conjunto das partes a cortar ou a armazenar que são designadas por *itens*. Durante o processo de corte ou de empacotamento são construídos padrões que são combinações geométricas dos itens nos objetos.

Como é percetível existe uma forte relação entre os problemas de corte e os problemas de empacotamento, que resulta da dualidade entre o material sólido e o espaço. O problema de empacotamento pode ser visto como um problema de cortar o espaço vazio dos objetos em partes do espaço que são ocupadas por itens, enquanto o problema de empacotamento pode ser abordado como o problema de empacotar o espaço ocupado pelos itens, no espaço ocupado pelos objetos.

Os problemas de corte e empacotamento têm sido muito estudados nas últimas décadas. Os primeiros estudos dos problemas de corte devem-se ao trabalho de Kantorovich (1960), cujo trabalho original foi publicado em russo em 1939, posteriormente seguem-se os trabalhos de Gilmore & Gomory (1961, 1963, 1965, 1967).

Desde então, surgiram na literatura trabalhos que abordam problemas similares, com a mesma estrutura lógica que os problemas de corte, no entanto sob designações diferentes, tais como: problemas de minimização dos desperdícios, de empacotamento, de empacotamento em tiras, problemas de mochila, de carregamento de veículos, de carregamento de paletes, de carregamento de contentores, entre outros. Estas denominações resultam diretamente das aplicações industriais em que os problemas tiveram origem.

2.2 Tipologias

Devido às várias denominações e ao crescente número de publicações na área do corte e empacotamento surgiu a necessidade de as classificar originando diferentes tipologias, nomeadamente a proposta por Dyckhoff (1990) e mais recentemente a apresentada por Wäscher et al. (2007). Entende-se por tipologia a organização sistemática de objetos em categorias homogéneas, baseando-se num conjunto de critérios de caracterização.

2.2.1 Tipologia de Dyckhoff (1990)

Dyckhoff (1990) introduziu quatro *critérios principais* de acordo com os quais os problemas de corte e empacotamento são categorizados. Da combinação das quatro características principais resultam 96 tipos de problemas. Os quatro critérios, com os respetivos tipos possíveis são agora definidos.

1. Dimensão

(1) Uma dimensão

(2) Duas dimensões

(3) Três dimensões

(N) N- dimensões, com N>3

2. Tipos de atribuição

(B) Todos os objetos e uma seleção dos itens

(V) Uma seleção dos objetos e todos os itens

3. Características dos objetos

(O) Um objeto

(I) Formas idênticas

(D) Formas diferentes

4. Características dos itens

(F) Poucos itens (de diferentes formas)

(M) Muitos itens de muitas formas diferentes

(R) Muitos itens com relativamente poucas formas diferentes (não congruentes)

(C) Formas congruentes

O primeiro critério é a *dimensão*, que representa o número mínimo de dimensões geométricas necessárias para descrever um padrão completamente. O segundo critério refere-se ao *tipo de atribuição dos itens aos objetos*; este critério engloba dois tipos, um em que uma seleção dos itens tem de ser atribuída aos objetos (B) e outro em que todos os itens têm de ser atribuídos aos objetos selecionados (V). O critério número três representa as *características dos objetos*,

i.e., se existe apenas um objeto (O), se todos os objetos têm forma idêntica (I), ou se os objetos têm todos formas diferentes (D). O último critério *caracteriza os itens*, i.e., se existem poucos itens e com diferentes formas (F), se existem muitos itens com muitas formas diferentes (M), ou se são considerados muitos itens com poucas formas diferentes (R) e por último, se os itens têm formas congruentes (C).

Os diferentes tipos de problemas são caracterizados por um tuplo quarto, com os respetivos símbolos a/b/c/d. Por exemplo, 2/B/O/F representa o tipo de problemas de corte bidimensionais, em que se pretende empacotar uma seleção de poucos itens num único objeto.

2.2.2 Tipologia de Wäscher et al. (2007)

Segundo Wäscher et al. (2007), a tipologia apresentada por Dyckhoff (1990) forneceu inicialmente um excelente instrumento de organização e categorização da literatura nova e existente dos problemas de corte, mas no entanto com o decorrer dos anos algumas insuficiências desta tipologia tornaram-se evidentes.

Deste modo, Wäscher et al. (2007) apresentam uma tipologia aperfeiçoada, baseando-se nas ideias originais de Dyckhoff (1990), mas introduzindo novos critérios de categorização e definindo diferentes categorias de problemas.

Wäscher et al. (2007) utilizaram cinco critérios para a definição dos tipos de problemas de corte: (i) a dimensão, (ii) o tipo de atribuição dos itens aos objetos, (iii) as características dos objetos, (iv) as características dos itens e (v) as formas dos itens.

Com a combinação dos critérios referidos, são definidos três tipos de problemas, os *básicos*, os *intermédios* e os *refinados*.

Os problemas do tipo básico são definidos através da combinação dos critérios:

tipo de atribuição dos itens aos objetos (ii) e características dos itens (iv). O critério tipo de atribuição tem em conta o objetivo final, ou seja, se é pretendida uma maximização do *output* ou uma minimização do *input*, enquanto o critério que caracteriza os itens distingue três casos: itens idênticos (todos são da mesma forma e tamanho), itens pouco heterogéneos (a procura dos itens é relativamente grande) e itens muito heterogéneos (poucos itens têm a mesma forma e tamanho, os itens são tratados como elementos individuais).

Tendo por base esta combinação de critérios, os autores definiram seis problemas básicos de corte e empacotamento: *problema de empacotamento de itens idênticos, problema de afetação, problema de mochila, problema de dimensão aberta, problema de corte e problema de empacotamento.* É apresentado um esquema sobre a obtenção dos tipos de problemas básicos na figura 2.1.

Com a finalidade de definir de um modo mais homogéneo os tipos de problemas, os problemas do tipo básico são divididos em problemas do tipo intermédio, sendo os problemas do tipo intermédio obtidos através da combinação dos problemas do tipo básico com o critério que caracteriza os objetos (iii). Os problemas do tipo refinado são definidos pelos problemas do tipo intermédio e pelos critérios dimensão (i) e forma dos itens (v). O critério dimensão distingue os problemas de uma, duas e três ou mais dimensões, enquanto o critério forma dos itens distingue os itens com formas regulares (retângulos, círculos, cilindros) dos itens com formas irregulares.

2.3 Problemas de corte e empacotamento bidimensionais

Os problemas de corte e empacotamento abordados nesta tese tem origem na indústria do mobiliário, e consistem no corte de placas de madeira ou aglomerado

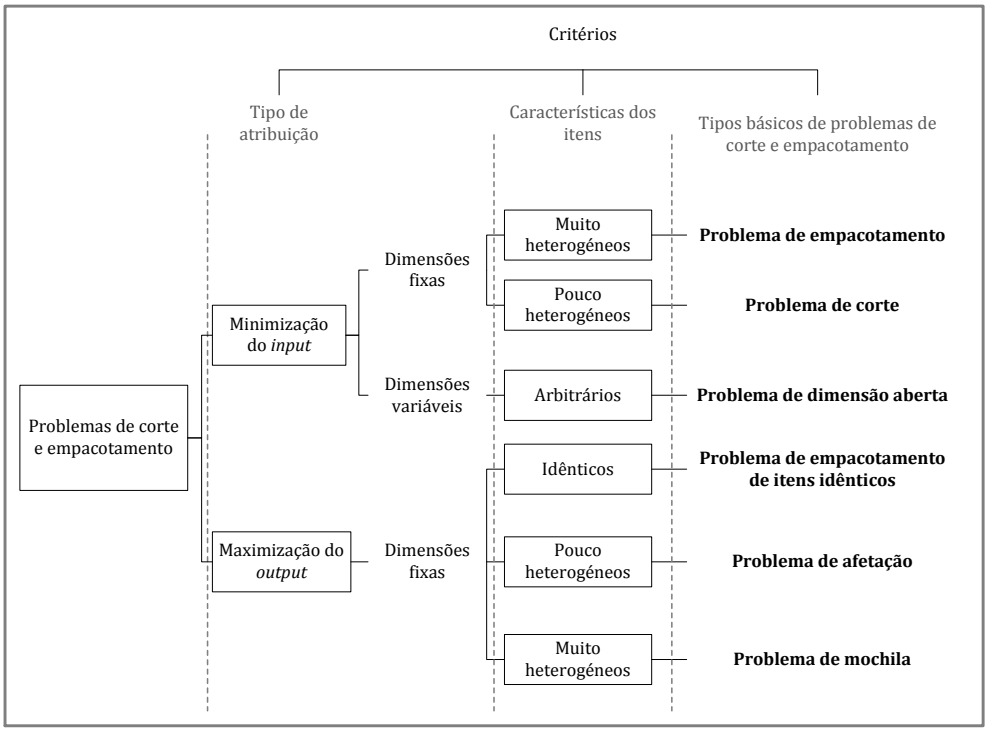


Figura 2.1: Problemas de corte e empacotamento segundo Wäscher et al. (2007).

em peças (itens) que serão utilizadas na produção de móveis. As placas são retangulares assim como as peças que serão cortadas para a construção do mobiliário e assume-se que a espessura dos itens e da placa é igual. Trata-se então de um problema de corte em que o número de dimensões espaciais no espaço euclideano é dois.

Uma combinação geométrica de itens na placa designa-se por *padrão de corte* para o problema de corte ou *padrão de empacotamento* para o problema de empacotamento. Na indústria do mobiliário, habitualmente, são apenas permitidos padrões de corte do tipo *ortogonal*, isto é, os cortes são efetuados paralelamente a um dos lados da placa, sendo os problemas em que é permitida qualquer ori-

entação dos cortes nos objetos denominados por *não - ortogonais*. Na figura 2.2 são apresentados dois tipos de padrão de corte, um ortogonal (a) e outro não - ortogonal (b).

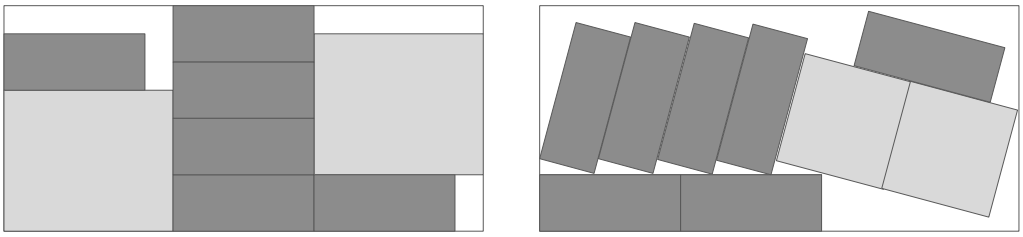


Figura 2.2: Padrão de corte ortogonal (a) e padrão de corte não-ortogonal (b).

Aos padrões de corte ortogonais podem também estar associados condicionalismos técnicos, relacionados com as ferramentas de corte, que obrigam a que os cortes sejam realizados de uma aresta à aresta oposta da placa, como é exemplo o corte de folhas de papel numa guilhotina, sendo os padrões de corte deste tipo designados por *guilhotinados*. Quando esta restrição não se aplica, os padrões de corte denominam-se por *não - guilhotinados*. Um exemplo deste tipo de padrões de corte é dado na figura 2.3.

Nos problemas considerados nesta tese, além da restrição de guilhotina, o número de *estágios* (conjuntos de cortes com a mesma orientação - horizontal ou vertical) no qual uma placa pode ser cortada é também limitado a dois ou três.

No problema de *dois-estágios*, os itens são obtidos através de um conjunto de cortes horizontais, dividindo a placa em tiras, seguido de um conjunto de cortes verticais, separando os diferentes itens. Se for permitido um conjunto de cortes adicionais de modo a separar os itens do desperdício, então o problema é um problema de *dois-estágios não-exato* (em oposição ao caso exato, onde todos os

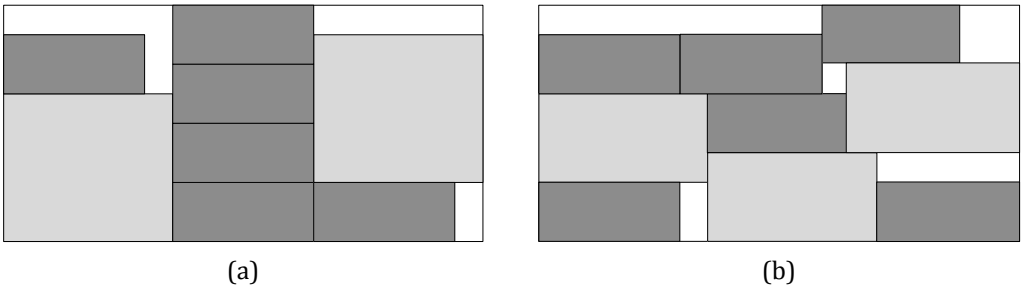


Figura 2.3: Padrão de corte guilhotinado (a) e padrão de corte não-guilhotinado (b).

itens numa tira têm a mesma altura). Na figura 2.4 apresentam-se dois padrões de corte, um para o problema de *dois-estágios exato* (a) e o outro para o problema de *dois-estágios não-exato* (b); os retângulos brancos representam o desperdício e os retângulos a sombreado representam os itens.

O problema de *três-estágios* é também considerado. Neste problema, é permitido um terceiro conjunto de cortes: depois da placa ser cortada em tiras (no primeiro estágio), cada tira é cortada em pilhas (segundo estágio) e finalmente os itens em cada pilha são separados por um terceiro conjunto de cortes (terceiro estágio). Tal como no problema de dois-estágios, são considerados os casos exato e não-exato. Para o caso exato, todos os itens na pilha têm a mesma largura, enquanto para o caso não-exato é permitido um conjunto de cortes verticais adicionais de modo a separar os itens do desperdício. Em ambos os casos, é considerada a versão restrita do problema de três-estágios, na qual em cada tira existe sempre uma pilha constituída por apenas um item, que define a altura da tira. Na figura 2.5 são apresentados dois exemplos para o problema de corte de três-estágios, em (c) para o caso exato e em (d) para o caso não-exato.

Existem autores que não fazem a distinção entre cortes exatos e não-exatos nos estágios, sendo que o corte adicional para separar itens de desperdício é

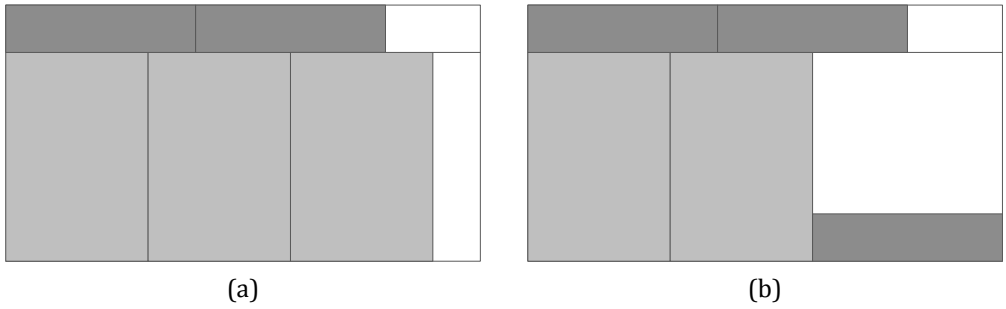


Figura 2.4: Padrão de corte de dois-estágios exato (a) e padrão de corte de dois-estágios não-exato (b).

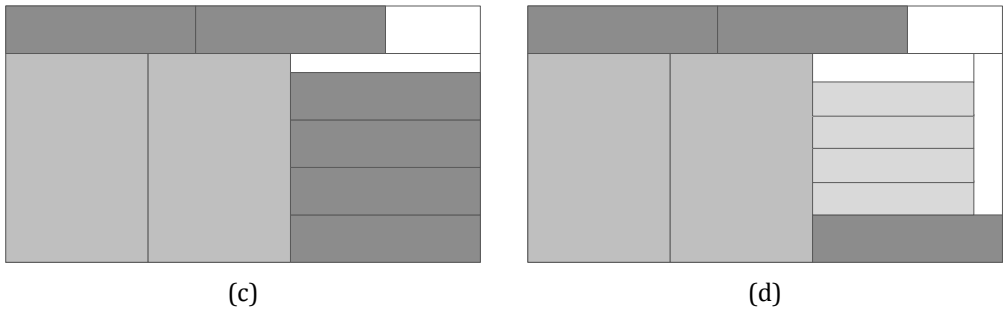


Figura 2.5: Padrão de corte de três-estágios exato (c) e padrão de corte de três-estágios não-exato (d).

considerado como sendo mais um estágio. Deste modo, para estes autores, o problema de dois-estágios não-exato é um problema de três estágios e o problema de três estágios não-exato é um problema de quatro estágios. Neste trabalho é considerada a distinção entre um corte que dá origem a dois itens, de um corte que produz um item e desperdício, sendo então considerados os problemas de dois-estágios exato e não-exato e os problemas de três-estágios exato e não exato.

De acordo com a tipologia de Dyckhoff (1990), os problemas a abordar nesta

tese são classificados como 2/V/I/R e 2/V/D/M, problema de corte a duas dimensões e problema de empacotamento a duas dimensões, respetivamente.

Segundo a tipologia de Wäscher et al. (2007) os problemas a tratar neste trabalho são problemas de corte bidimensionais com placas retangulares únicas (SSSCSP - Single Stock Size Cutting Stock Problems) e problemas de empacotamento a duas dimensões com placas retangulares únicas (SBSBPP - Single Bin Size Bin Packing Problems) com restrições adicionais relacionadas com os tipos de cortes permitidos.

2.4 Modelos e métodos de resolução

Nesta secção apresenta-se uma revisão bibliográfica dos métodos de resolução utilizados para os problemas de corte e empacotamento bidimensionais.

Em primeiro lugar é efetuada uma revisão dos trabalhos que utilizam geração de colunas. Esta metodologia iniciou-se com o trabalho pioneiro de Gilmore & Gomory (1965) e consiste na utilização de modelos de programação (linear) inteira em que as variáveis de decisão estão associados a padrões de corte. Dado o número exponencial de padrões de corte, é frequente a resolução basear-se no método de geração de colunas em que o subproblema é um problema de corte a duas dimensões numa única placa; este problema é também de difícil modelação e resolução devido aos seus aspectos geométricos. Deste modo, é apresentada uma revisão bibliográfica dos modelos que utilizam geração de colunas para resolver o problema de corte bidimensional, assim como dos métodos e modelos para resolver o subproblema associado à geração de colunas.

Na subsecção 2.4.2 é apresentada uma revisão dos modelos de programação inteira existentes na literatura para os problemas de empacotamento bidimensionais.

Por fim, a última subsecção é dedicada às abordagens heurísticas que têm vindo a ser utilizadas para resolver problemas de corte e de empacotamento a duas dimensões.

No capítulo 4 é estudado o problema integrado de corte bidimensional e dimensionamento de lotes, sendo a revisão bibliográfica referente a este problema apresentada no início desse capítulo.

2.4.1 Geração de colunas

A geração de colunas é um método de resolução que é tipicamente utilizado para resolver problemas de programação linear de grande dimensão, ou também para obter bons limites inferiores para problemas de programação inteira.

Os primeiros trabalhos que utilizam geração de colunas são de Ford & Fulkerson (1958), Dantzig & Wolfe (1960) e Gilmore & Gomory (1961, 1963).

Na geração de colunas um dado problema original é resolvido através da resolução de uma sequência de problemas de menor dimensão. Para problemas de programação inteira as restrições de integralidade são relaxadas e é criado um problema de programação linear com uma versão restrita do problema original (não são consideradas explicitamente todas as variáveis/colunas); este problema é denominado de problema mestre restrito (PMR). A geração de colunas obtém a solução ótima para a relaxação linear do problema original através da troca de informações entre o PMR e os subproblemas, sendo que estes últimos podem ser resolvidos por qualquer algoritmo de otimização, dependendo do problema.

A geração de colunas é um processo iterativo, sendo adicionadas novas colunas ao PMR obtidas através da resolução dos subproblemas. Em todas as iterações o PMR é resolvido otimamente e os seus valores ótimos duais são utilizados na função objetivo dos subproblemas, que determinam se ainda existem colunas atrativas para inserir no PMR; sendo assim a geração de colunas vai alternando entre

o PMR e os subproblemas.

O trabalho de Gilmore & Gomory (1965) foi pioneiro nos modelos orientados ao padrão nos problemas de corte unidimensional. Tal como no problema a uma dimensão, Gilmore & Gomory (1965) consideram para o problema bidimensional uma variável para cada padrão de corte admissível, mas no entanto constatam que é difícil resolver o problema desta forma, uma vez que existe um número elevado de padrões de corte possíveis, isto é, um número elevado de variáveis de decisão. Sendo assim, utilizam a geração de colunas para determinar, em cada passo, padrões de corte promissores (colunas) que permitam melhorar a solução atual.

Considere-se a matriz A, com n linhas e um número elevado m de colunas, uma para cada padrão de corte de dois-estágios possível, os elementos a_{ij} $(i = 1, ..., n, j = 1, ..., m)$ indicam o número de vezes que o item do tipo i ocorre no padrão de corte j. O vector coluna X representa as variáveis de decisão, uma para cada coluna de A e a variável inteira x_j representa o número de vezes que o padrão j deve ser utilizado. A procura do item i é representada por d_i. Pretende-se a minimização da soma do número de vezes que cada padrão de corte é utilizado, isto é, a minimização do número de placas utilizadas.

O modelo de programação inteira correspondente é o seguinte:

$$\text{Minimizar} \quad \sum_{j=1}^{m} x_j \tag{2.1}$$

$$\text{Sujeito a:} \quad \sum_{j=1}^{m} a_{ij} x_j \geq d_i, \qquad i = 1, ..., n \tag{2.2}$$

$$x_j \geq 0 \text{ e inteiro}, \qquad j = 1, ..., m \tag{2.3}$$

Devido ao imenso número de colunas da matriz A, a resolução deste problema

torna-se bastante difícil. Para contornar esta dificuldade, tal como no método proposto para o problema de corte unidimensional por Gilmore & Gomory (1961, 1963), Gilmore & Gomory (1965) aplicam a técnica de geração de colunas. A condição de integralidade é relaxada e resolve-se um problema de programação linear. A cada iteração, a informação dual fornecida pelo problema mestre (2.1) - (2.3) é transferida para o subproblema, que determina qual o padrão de corte (coluna) mais atrativo para fazer parte da solução.

O subproblema consiste então na criação de padrões de corte, isto é, em combinações geométricas dos itens numa única placa. Salienta-se que através da geração de colunas é apenas obtida a solução da relaxação linear do problema original. No entanto com base na solução da relaxação linear podem ser obtidas soluções para o problema inteiro, por exemplo através de arredondamento.

No subproblema é considerada a alocação dos itens a uma única placa e cada item é caracterizado não só pelo seu tamanho, mas também por um determinado valor (valor da variável dual associada ao item no problema mestre). Pretende-se encontrar a seleção de itens para a qual o valor dos itens cortados seja máximo. Segundo a tipologia de Wäscher et al. (2007), e tendo em conta que se pretende a maximização do *output*, este problema é classificado como de empacotamento de itens idênticos, de atribuição ou de mochila, dependendo das características dos itens.

Existem na literatura vários trabalhos que abordam este problema, que, além de considerarem as restrições geométricas, têm ainda em consideração restrições relacionadas com o número de vezes que cada tipo de item pode ser colocado no padrão de corte, caso em que o padrão é denominado por *restrito*, enquanto que quando esta limitação não existe, o padrão é intitulado *não restrito*.

Apresenta-se em seguida um levantamento bibliográfico de alguns métodos de

resolução para o subproblema. Alguns destes trabalhos não se referem especificamente à geração de colunas, mas no entanto os seus métodos podem ser utilizados para gerar padrões de corte atrativos para o problema mestre.

Subproblemas não restritos

Na abordagem clássica proposta por Gilmore e Gomory, o subproblema é resolvido através de uma sequência de dois tipos de problemas inteiros de mochila, de modo a encontrar um padrão de corte do tipo dois-estágios não-exato. No total são resolvidos $n+1$ problemas de mochila: nos primeiros n problemas de mochila são geradas tiras horizontais e no último as tiras selecionadas são agrupadas verticalmente de modo a formarem um padrão de corte.

Considere-se π_i, $i = 1, .., n$, o valor das variáveis duais associados às restrições de procura dos itens no problema mestre. O primeiro tipo de problema de mochila é resolvido para cada item do tipo $i, i = 1, .., n$. Para cada altura h_i calcula-se Z_i^*, o ótimo obtido através da alocação dos itens com altura h_j e largura w_j com $h_j \leq h_i$ numa tira de altura h_i e largura W. Deste modo, com a solução dos n problemas de mochila do tipo (2.4)-(2.7) serão construídas n tiras com altura h_i $i = 1, .., n$.

$$\text{Maximizar } Z_i^* = \sum_{j \in I^*} \pi_j b_j \tag{2.4}$$

$$\text{Sujeito a:} \quad \sum_{j \in I^*} w_j b_j \leq W - w_i, \qquad \forall j \in I^* \tag{2.5}$$

$$b_j \geq 0 \text{ e inteiro}, \qquad \forall j \in I^* \tag{2.6}$$

$$I^* = \{h_j : h_j \leq h_i, i = 1, ..., n\} \tag{2.7}$$

Uma vez calculadas todas as tiras possíveis e o respetivo valor Z_i^*, o problema de mochila (2.8) - (2.10) determina quais as tiras que constituem o padrão de

corte do tipo dois-estágios não-exato.

$$\text{Maximizar} \quad \sum_{i=1}^{n} Z_i^* y_i \tag{2.8}$$

$$\text{Sujeito a:} \quad \sum_{i=1}^{n} h_i y_i \leq H, \qquad i = 1, ..., n \tag{2.9}$$

$$y_i \geq 0 \text{ e inteiro}, \qquad i = 1, ..., n \tag{2.10}$$

No mesmo trabalho Gilmore e Gomory propõem também uma segunda abordagem ao problema de corte bidimensional com padrões do tipo dois-estágios não-exato, aplicando um misto de geração de colunas com a decomposição de Dantzig e Wolfe.

O problema é do mesmo modo tratado em duas fases. Na primeira fase, considera-se o corte de tiras com altura h_i (altura dada pelo item i), das placas com altura H. Na segunda fase, cada tira de altura h_s, $s = 1, .., n$, é considerada e obtêm-se a partir destas os itens pedidos.

O problema é formulado do seguinte modo:

$$\text{Maximizar} \quad \sum_{j} y_j^0 \tag{2.11}$$

$$\text{Sujeito a:} \quad Ax^* \geq D^*, \qquad i = 1, ..., n \tag{2.12}$$

$$y_i \geq 0 \text{ e inteiro}, \qquad i = 1, ..., n \tag{2.13}$$

A matriz A é constituída por $2n$ linhas e $n + 2$ submatrizes A_s, $s = 1, .., n$. A primeira parte da matriz A é formada por A_0, as suas colunas representam o corte das placas em tiras, sendo os primeiros n elementos inteiros não negativos $a_1, ..., a_n$, que satisfazem $\sum_{i=1}^{n} a_i h_i \leq H$, os restantes n elementos são todos zeros. Nas submatrizes $A_s, s = 1, .., n$, as tiras de altura h_s são cortadas nos itens pedidos, as colunas destas matrizes são formadas por zeros nas primeiras n linhas

exceto na $s-$ésima linha com valor -1, com inteiros b_i não negativos nas linhas $n+i, i = 1, .., s$, que satisfazem $\sum_{i=1}^{n} b_i w_i \leq W$, e zeros nas restantes linhas. A última submatriz A_{n+1} é formada por zeros nas primeiras n linhas e a matriz identidade nas últimas n linhas.

O vetor D^* é constituído por n linhas com valor zero e n linhas com os valores das procuras de cada item.

As variáveis y_1^0, y_2^0... representam o número de vezes que cada um dos diferentes padrões de corte formados pelas diferentes tiras é utilizado. Enquanto as variáveis y_1^s, y_2^s... representam o número de vezes que os padrões de corte para as tiras de altura h_s são usados. As variáveis de folga são representadas por μ.

Deste modo, a formulação (2.11)-(2.13) tem a seguinte forma:

Tabela 2.2: Formulação.

	-1 , ..., -1	0				y_0^*		0
A_0	0	-1 , ..., -1		0	0			
		0	...	-1 , ..., -1		y_1^*		
							=	D^*
	A_1							
		A_2						
0	0	0		A_n	$-I$	y_n^*		
						μ		0

As primeiras n equações garantem que na segunda fase são usadas, exatamente, as tiras produzidas na primeira fase.

O segundo conjunto de restrições garante que a procura dos itens é satisfeita, através do corte das tiras.

Em Herz (1972) é apresentado um algoritmo recursivo (árvore de pesquisa)

para o problema guilhotinado sem limites no número de estágios. O algoritmo consiste em tentar todos os primeiros cortes possíveis e selecionar apenas o que tiver valor máximo para os dois retângulos gerados. Para reduzir o número de cortes testados são definidos conjuntos de discretização, constituídos pelas combinações lineares dos tamanhos dos itens e por restrições de simetria. Os padrões cujos cortes respeitam estas considerações foram denominados de "dissecações homogéneas". A memorização é também utilizada no algoritmo de modo a evitar a pesquisa de retângulos parciais já explorados. O algoritmo recursivo é comparado com o algoritmo iterativo de Gilmore & Gomory (1967), concluindo-se que o recursivo é 20% mais rápido.

No entanto, segundo Beasley (1985), o algoritmo de Herz é muito dependente de um limite superior para o valor do padrão de corte ótimo para todos os retângulos, que para o caso em que se pretende a maximização da área cortada será simplesmente o valor da área do retângulo, mas para os problemas com objetivo diferente poderá ser de difícil resolução.

Beasley (1985) apresenta algoritmos heurísticos e ótimos baseados em programação dinâmica para os problemas guilhotinados com e sem limite no número de estágios. Para o problema com número de estágios limitado, propõe um modelo de programação dinâmica recursiva e aponta um erro no modelo de programação dinâmica proposto por Gilmore & Gomory (1967) para o mesmo problema, justificando que o erro reside na não consideração de todas as orientações possíveis para o primeiro corte. A eficiência do modelo de programação dinâmica é melhorada através da consideração de apenas padrões de corte normais, introduzidos por Herz (1972), Christofides & Whitlock (1977). No entanto, o modelo de programação dinâmica torna-se computacionalmente inviável se as dimensões da placa forem muito grandes quando comparadas com os tamanhos dos itens. Para este caso, o modelo que garante a solução ótima é alterado para um método

heurístico garantindo a viabilidade computacional. Para o problema guilhotinado geral (sem restrição no número de estágios) fazem uma adaptação no modelo de programação dinâmica usado para o problema com estágios. Baseando-se no modelo de programação dinâmica conjugado com alterações nos conjuntos de cortes admissíveis, deixando de garantir a solução ótima, propõe uma formulação de programação dinâmica recursiva heurística.

Em Hifi (2001) são propostos modelos exatos para os problema de dois e três-estágios, considerando os casos exato e não-exato. A rotação dos itens é permitida para o problema de dois-estágios enquanto para o de três-estágios são considerados ambos os casos, orientação fixa ou permitindo a rotação. O modelo utilizado para o problema de dois-estágios é uma extensão ao modelo proposto por Gilmore & Gomory (1965). Para o problema de três-estágios apresenta um algoritmo baseado numa pesquisa em grafo combinado com programação dinâmica.

Alvarez-Valdes et al. (2002) apresentam heurísticas para quatro versões do problema, i.e., para os problemas em que os valores dos itens são iguais ou diferentes das suas áreas e para os problemas restritos e não-restritos. Começam por desenvolver heurísticas construtivas que utilizam como blocos de construção para procedimentos mais complexos e utilizam também uma heurística *GRASP* (*greedy randomised adaptive search procedure*). Apresentam também uma pesquisa tabu e um procedimento de "*path relinking*".

Subproblemas restritos

Christofides & Whitlock (1977) apresentaram o primeiro método exato de pesquisa em árvore para o problema em que existem restrições no número de tipos de itens cortados. Consideram o problema guilhotinado sem restrição no número de cortes. O tamanho da árvore de pesquisa é limitado através da derivação

e imposição de condições necessárias para que o padrão de corte seja ótimo e utilizando limites superiores durante a pesquisa. Estes limites são calculados através de programação dinâmica para o problema não restrito e de um método de avaliação do nó baseado no problema de transportes. Consideram também os efeitos da simetria e os efeitos da ordem pela qual os cortes são efetuados, tendo por finalidade a não repetição de padrões de corte, isto é, padrões constituídos pelos mesmos itens, mas em posições diferentes. Introduzem o conceito de padrão de corte normal, no qual os cortes são efetuados em localizações pré-calculadas, correspondendo a combinações lineares das larguras e comprimentos dos itens.

Em Christofides & Hadjiconstantinou (1995) é apresentado um melhoramento ao modelo de pesquisa em árvore proposto por Christofides & Whitlock (1977). É também proposto um algoritmo de árvore de pesquisa e considerado um limite superior mais apertado, que resulta da relaxação do espaço da formulação de programação dinâmica do problema original.

Wang (1983) propõe, para o problema restrito guilhotinado sem restrição no número de estágios, dois algoritmos combinatórios, que geram padrões de corte através de construções horizontais e verticais sucessivas. O algoritmo gera incrementalmente retângulos, considerando os itens iniciais e novos retângulos que resultam de combinações anteriores de itens. É utilizado um limite máximo de percentagem de desperdício, e limites de erro que medem o quão perto os desperdícios do padrão gerado estão do desperdício da solução ótima. Um aperfeiçoamento ao algoritmo de Wang é proposto por Oliveira & Ferreira (1990), modificando o critério de aceitação das combinações dos itens através da redefinição do limite de percentagem de desperdício, usando o método de Gilmore & Gomory (1967). Posteriormente em Parada, Alvargenga & Diego (1995) é proposto um novo melhoramento ao modelo de Wang considerando as alterações propostas por Oliveira & Ferreira (1990). Neste trabalho para a geração dos retângulos é utilizada a re-

presentação no grafo E/OU, sendo a pesquisa no referido grafo efetuada de forma informada, isto é utilizando diferentes funções de avaliação para guiar a pesquisa. Em Viswanathan & Bagchi (1993) é também utilizado o algoritmo de Wang num algoritmo baseado em pesquisa em árvore.

O método de pesquisa por arrefecimento simulado foi também utilizado por Parada et al. (1998) para resolver o problema guilhotinado restrito sem restrição no número de estágios. Foi utilizada uma árvore binária para a representação do padrão de corte, em que os nós representam retângulos que estão identificados por dois pares ordenados, o primeiro indicando as coordenadas do canto superior esquerdo (coordenadas de origem) e o segundo indicando as coordenadas do canto inferior direito. A estrutura de representação em árvore facilita a geração de soluções vizinhas; a função de avaliação é baseada nos desperdícios e é calculada através de uma exploração da árvore. O método foi comparado com os algoritmos de Wang (1983), Oliveira & Ferreira (1990) e Parada, Alvargenga & Diego (1995).

No trabalho de Parada et al. (2000) é realizado um estudo comparativo do algoritmo de Wang, do procedimento de Oliveira & Ferreira (1990), do método de Parada, Alvargenga & Diego (1995) que utiliza o algoritmo de Wang e uma representação no grafo E/OU, do método de pesquisa por arrefecimento simulado de Parada et al. (1998) e de um algoritmo genético proposto por Parada, Muñoz & Gómes (1995). Para o efeito geraram 1000 instâncias aleatoriamente, concluindo que o método de pesquisa por arrefecimento simulado e o algoritmo genético resolvem rapidamente todas as instâncias com uma percentagem de desperdício que não ultrapassa os 20%. Concluem também que o algoritmo de Wang encontra muitas vezes o ótimo para instâncias com poucas combinações de itens. Referem ainda, que embora os modelos de Oliveira & Ferreira (1990) e de Parada, Alvargenga & Diego (1995) tenham obtido melhores soluções que o algoritmo de Wang, para instâncias com um grau combinatório baixo dos itens, o algoritmo de Wang

obtém menores percentagens de desperdício. Comparando o método de pesquisa por arrefecimento simulado com o algoritmo genético conclui-se que o primeiro é mais rápido enquanto que o último consegue soluções de melhor qualidade.

Um algoritmo aproximado e um algoritmo exato são propostos por Hifi (2001) para os problemas de dois-estágios exato e não-exato, cujos valores dos itens são iguais ou diferentes dos valores das suas áreas. O algoritmo aproximado tem por base a programação dinâmica clássica, recorrendo à decomposição do problema em problemas de mochila similares aos utilizados por Gilmore & Gomory (1965). O modelo exato é conseguido através da combinação do método de partição com programação dinâmica.

Em Lodi & Monaci (2003) são propostos dois modelos de programação linear inteira, envolvendo um número polinomial de variáveis e restrições. Num modelo os itens são considerados individualmente, ignorando-se a existência de itens com o mesmo tamanho, enquanto que no outro modelo estas características dos itens são levadas em consideração. É abordado o problema de dois-estágios não-exato, mas os modelos podem ser estendidos para outras variantes do problema podendo considerar-se o problema de dois-estágios exato, o problema não restrito e a rotação dos itens. Os limites superiores obtidos através da relaxação linear dos modelos são comparados com os limites superiores clássicos obtidos através do modelo de geração de colunas proposto por Gilmore & Gomory (1961), concluindo-se que os obtidos através da relaxação linear são mais fracos. Os modelos são ainda comparados com o modelo exato de Hifi (2001).

Em Hifi (2004) é apresentada uma extensão ao modelo de Hifi (1997) para o problema não restrito. É proposto um algoritmo híbrido através da combinação de uma pesquisa em profundidade usando "*hill-climbing*"e programação dinâmica.

Um método de partição e geração de colunas com cortes é proposto por Belov & Scheithaeur (2006) para o problema de dois-estágios não-exato com orienta-

ção fixa dos itens. Os autores afirmam que os modelos de geração de colunas de Gilmore & Gomory (1961, 1963, 1965) têm relaxações lineares muito fortes e que permitem obter bons limites inferiores para soluções de programação linear. No modelo proposto, cada nó do método de partição e geração de colunas é fortalecido através de cortes de Chvátal-Gomory e cortes inteiro-mistos e a partição é efetuada nas variáveis da formulação de Gilmore & Gomory (1965). Comparam o modelo proposto com os modelos de Hifi (2001) e Lodi & Monaci (2003).

Em Alvarez-Valdes et al. (2007) desenvolvem dois algoritmos baseados na metodologia "*greedy randomized adaptative search procedure*" (*GRASP*), um "orientado ao item" e outro "orientado à tira"; com as melhores soluções destes modelos criam um conjunto de soluções elite às quais é aplicado um algoritmo "*path relinking*".

Um algoritmo exato e duas heurísticas são propostos em Cui (2008) para gerar padrões de corte do tipo três-estágios homogéneos, isto é, padrões em que as pilhas são constituídas por itens do mesmo tipo. O modelo baseia-se no método de partição e avaliação sucessivas combinado com programação dinâmica.

Em Chen (2008) é proposto um algoritmo recursivo para o problema guilhotinado sem limite no número de estágios, com recurso a algumas regras heurísticas.

Grafo E/OU

Morabito & Arenales (1994) propõem uma abordagem para resolver o subproblema de criar padrões de corte para o problema de corte a duas dimensões. Consiste na representação do espaço de soluções como caminhos completos num grafo E/OU. Este método foi utilizado para resolver problemas do tipo restrito e não restrito.

No grafo E/OU os nós representam retângulos, enquanto os arcos-E correspon-

dem aos cortes sobre os retângulos, ligando um nó aos outros (retângulos obtidos após o corte), definindo uma relação entre um nó e um subconjunto de nós, e os arcos-OU representam as diferentes opções para ramificar um nó, emergindo muitos arcos de um mesmo nó.

Os padrões de corte são gerados examinando-se todas as possibilidades alternativas de corte (arcos-OU), sendo uma delas reproduzir o próprio retângulo (corte-0) ao qual nenhum outro corte será efetuado, indicando o fim do processo de corte (nós finais).

Em cada retângulo pode ser efetuado um número alternativo de cortes, levando a diferentes padrões de corte. Se forem investigadas todas as alternativas em cada retângulo incluindo o corte-0, seriam gerados todos os padrões de corte possíveis. Um caminho completo no grafo corresponde a um padrão de corte. O caminho inicia-se num nó inicial, que representa a placa, e a partir daí escolhem-se os arcos-E e os cortes-0 até que todos os nós sejam finais.

O padrão de corte ótimo resulta da seleção do caminho completo mais valioso no grafo E/OU. No entanto, de um modo geral, a enumeração completa de todos os nós é computacionalmente inviável. De modo a evitar que diferentes caminhos gerem padrões de corte equivalentes aplicam-se regras de redução de simetria, exclusão e ordenação de cortes.

Para a pesquisa no grafo é utilizada uma abordagem híbrida, combinando "*backtracking*" com "*hill-climbing*"; recorrem também ao uso de limites inferiores e superiores assim como de heurísticas.

A utilização do grafo E/OU como abordagem para a construção de padrões de corte foi utilizada em diferentes tipos de problemas de corte bidimensional: em Arenales & Morabito (1995) é estudado o problema não restrito e não-guilhotinado, em Morabito & Arenales (1996) é abordado o problema restrito e guilhotinado e em Morabito & Arenales (1994) é tratado o problema a três

dimensões.

Mais recentemente foi utilizado por Morabito & Pureza (2010) para o problema restrito guilhotinado sem restrição no número de estágios. Nesta abordagem o grafo E/OU surge combinado com uma relaxação do espaço da formulação de programação dinâmica, com o método de otimização do subgradiente e uma heurística que transforma soluções inadmissíveis em soluções admissíveis.

Até aqui foi apresentada a abordagem clássica de geração de colunas de Gilmore & Gomory (1965) para o problema de corte assim como um conjunto de métodos existentes na literatura para resolver o subproblema. No entanto, existem outros trabalhos na literatura, posteriores ao trabalho de Gilmore & Gomory (1965), que também recorrem à geração de colunas. Apresentam-se em seguida alguns desses trabalhos.

Em Oliveira & Ferreira (1994) é apresentada uma abordagem diferente do modelo clássico de geração de colunas de Gilmore & Gomory (1965). Na técnica que propõem o subproblema não é resolvido até ao ótimo, utilizando antes uma heurística gulosa para a geração de padrões de corte. Somente quando a heurística não encontra um padrão com custos reduzidos negativos o subproblema é resolvido até à otimalidade através de programação dinâmica. Pretendem diminuir o tempo médio por iteração, embora com esta abordagem possam aumentar o número de iterações, sendo assim a eficiência desta abordagem está dependente do produto entre o número de iterações e o tempo por iteração. Os autores denominaram esta técnica de geração rápida de colunas. Para obterem a solução inteira, a solução da relaxação linear é arredondada para o inteiro abaixo e uma heurística calcula padrões de corte com os itens em falta. Efetuaram testes computacionais de modo a comparar as duas técnicas de geração de colunas e concluem que a geração rápida de colunas é cerca de duas vezes mais rápida que o modelo clássico

de geração de colunas.

Para o problema de corte de dois-estágios com procuras e valores muito variados, Riheme et al. (1996) apresentam também um modelo baseado em geração de colunas. O modelo proposto considera um novo problema relaxado, que é usado para construir padrões do tipo dois-estágios, que satisfaçam a maior parte das procuras. As procuras que ainda não estejam completamente satisfeitas são consideradas num problema residual, que é resolvido como um problema com procuras pequenas.

Vanderbeck (2001) também utilizou o método de geração de colunas para o problema de três-estágios, com várias restrições adicionais, relacionadas com o processo de corte na aplicação industrial onde o problema teve origem. O subproblema para a geração de padrões atrativos é também resolvido através de geração de colunas.

Em Alvarez-Valdes et al. (2002) é também utilizado o modelo clássico de geração de colunas proposto por Gilmore e Gomory. Neste trabalho o subproblema é resolvido heuristicamente através de uma hierarquia de três heurísticas de complexidade crescente, uma heurística construtiva, uma GRASP e por último, uma pesquisa tabu. A solução inteira é obtida através de diferentes métodos de arredondamento, arredondamento para cima, pesquisa e avaliação sucessivas truncada e a solução de um problema residual.

Uma heurística baseada em cobertura de conjuntos é utilizada para o problema de empacotamento em Monaci & Toth (2006). Nesta formulação é também utilizada a geração de colunas, no entanto como é considerado o problema de empacotamento, o problema mestre é equivalente a um problema de cobertura de conjuntos, uma vez que as procuras dos itens neste problema são iguais a um. O algoritmo proposto começa por calcular os limites inferiores de Martello & Vigo (1998) e de Boschetti & Mingozzi (2003*a*). Depois geram um conjunto de colunas

utilizando heurísticas construtivas propostas por Berkey & Wang (1987), as heurísticas *Alternate Directions* e *Floor ceiling* de Lodi et al. (1998) e a heurística *HBP* de Boschetti & Mingozzi (2003*a*). Cada uma das heurísticas é aplicada várias vezes, recorrendo-se a diferentes critérios de ordenação dos itens. Numa segunda fase, o problema de cobertura de conjuntos, que contém as colunas geradas pelas heurísticas construtivas, é resolvido através da heurística lagrangeana proposta por Caprara et al. (1999).

Outro modelo de geração de colunas é proposto por Pisinger & Sigurd (2007), também para o problema de empacotamento. O subproblema é resolvido heuristicamente e sempre que não é encontrado um padrão de corte com um custo reduzido negativo, o subproblema é então resolvido otimamente. Para que o subproblema seja resolvido exatamente dividem-no em dois problemas. O primeiro consiste num problema de mochila, que determina um subconjunto de itens com lucro máximo cuja soma das áreas não ultrapasse a área da placa. O segundo problema determina se é possível empacotar este subconjunto de itens numa placa, através da resolução de um modelo de programação em lógica com restrições que é resolvido através de um algoritmo recursivo. Neste trabalho propõem também um modelo de programação em lógica com restrições, no entanto o limite inferior obtido com a relaxação linear deste modelo é fraco, e além disso, os *solvers* gerais de programação linear e inteira têm dificuldade em resolver este modelo. Consideram também extensões ao modelo utilizado para resolver o subproblema, considerando outras restrições como cortes guilhotinados, posições relativas dos itens, posições fixas e placas irregulares, isto é, com defeito.

Em Puchinger & Raidl (2007) é também utilizada geração de colunas para o problema de empacotamento, mas para o problema em três-estágios. Tendo em conta que as procuras são iguais a um, o problema mestre é um problema de cobertura de conjuntos tal como em Pisinger & Sigurd (2007). O processo de gera-

ção de colunas é denominado de geração rápida de colunas, sendo o subproblema resolvido através de uma hierarquia de quatro métodos (primeiro é utilizada uma heurística gulosa, seguido de um algoritmo genético, que é seguido por um método exato para uma versão simplificada do subproblema, e apenas se não for encontrado nenhum padrão atrativo com os métodos anteriores, o subproblema é resolvido exatamente). A estabilização da geração de colunas é conseguida no modelo de partição e geração de colunas através de desigualdades duais ótimas Valério de Carvalho (2005).

Mais recentemente Cintra et al. (2008) utilizam também geração de colunas para os problemas de dois, três e quatro estágios permitindo que os itens tenham ou não orientação fixa. Para além do problema de corte a duas dimensões consideram também o problema de corte com múltiplas placas ou de dimensão aberta. O subproblema é resolvido através de programação dinâmica, usando a fórmula de recorrência proposta por Beasley (1985) combinada com a definição de pontos de discretização de Herz (1972). Depois de resolvida a relaxação linear, arredondam a solução para baixo, calculam uma instância residual com os itens para os quais a procura ainda não foi satisfeita, resolvem a sua relaxação linear e procedem de modo iterativo.

2.4.2 Afetação

Modelos compactos de programação inteira têm sido propostos para os problemas de empacotamento. Em Lodi, Martello & Monaci (2002) é proposto um modelo de programação binária para o problema de dois-estágios não-exato. O modelo é caracterizado por um número de variáveis e restrições polinomial em relação ao tamanho da instância. As variáveis de decisão estão relacionadas com a afetação dos itens às tiras, que constituem os níveis, e com a afetação destas às placas. A etapa de implementação é reduzida à introdução do modelo num *solver* de

programação inteira. Para a construção da formulação fazem as seguintes considerações: (i) o item empacotado mais à esquerda em cada tira é o item mais alto, (ii) a primeira tira em cada placa é a mais alta e (iii) os itens são ordenados por ordem decrescente de alturas. Assumem que estão disponíveis n tiras. A variável binária $y_i, i = 1, ..., n$, é igual a 1 se o item i inicializa a tira i, 0 caso contrário, a variável $x_{ij}, i = 1, ..., n-1; j > i$ é igual a 1 se o item j é empacotado no nível i. Do mesmo modo, em relação às placas, assumem que estão disponíveis n placas potenciais. A variável binária $q_k, k = 1, ..., n$, toma valor 1 se a tira k inicia a placa k e valor 0 caso contrário e a variável binária $z_{ki}, k = 1, ..., n-1; i > k$ é igual a 1 se a tira i é colocada na placa k e 0 caso contrário. A formulação é então a seguinte:

$$\text{Maximizar} \quad \sum_{k=1}^{n} q_k \tag{2.14}$$

$$\text{Sujeito a:} \quad \sum_{i=1}^{j-1} x_{ij} + y_j = 1 \qquad j = 1, ..., n \tag{2.15}$$

$$\sum_{j=i+1}^{n} w_j x_{ij} \leq (W - w_i) y_i \qquad i = 1, ..., n-1 \tag{2.16}$$

$$\sum_{k=1}^{i-1} z_{ki} + q_i = y_i \qquad i = 1, ..., n \tag{2.17}$$

$$\sum_{i=k+1}^{n} h_i z_{ki} \leq (H - h_k) q_k \qquad i = 1, ..., n \tag{2.18}$$

$$y_i \in \{0, 1\} \qquad i = 1, ..., n \tag{2.19}$$

$$x_{ij} \in \{0, 1\} \qquad i = 1, ..., n-1; j > i \tag{2.20}$$

$$q_k \in \{0, 1\} \qquad k = 1, ..., n \tag{2.21}$$

$$z_{ki} \in \{0, 1\} \qquad k = 1, ..., n-1; i > k \tag{2.22}$$

A função objetivo (2.14) minimiza o número de placas usadas. As restrições (2.15) garantem que cada item é colocado exatamente uma vez numa tira. As

restrições (2.16) asseguram que a largura da placa não é excedida; de igual modo as restrições (2.18) garantem que a altura da placa é respeitada. Por último, as restrições (2.17), tal como a restrição (2.15), não permitem que uma tira seja alocada a mais do que uma placa.

Em Puchinger & Raidl (2007), o modelo de Lodi, Martello & Monaci (2002) é estendido para o problema de três-estágios exato. Neste problema as variáveis de decisão estão associadas à afetação dos itens às pilhas, destas às tiras e por último a atribuição das tiras às placas. Neste trabalho são propostos dois modelos de programação inteira com variáveis binárias. O primeiro é denominado de restrito, uma vez que a altura de um nível é definida pela altura do seu item mais alto. No segundo modelo, designado não-restrito, a restrição da altura não é mais imposta, sendo esta definida pela pilha mais alta.

No problema restrito a variável binária α_{ji} $j = 1, ..., n, i = j, ..., n$, toma valor 1 se o item i está na pilha j e valor 0 caso contrário; do mesmo modo a variável binária $\beta_{kj}, k = 1, ..., n, j = k, ..., n$, é igual a 1 se a tira k contém a pilha j e por último a variável $\gamma_{lk}, l = 1, ..., n, k = l, ..., n$, toma valor 1 se a tira k está contida na placa l e 0 caso contrário.

$$\text{Maximizar} \quad \sum_{l=1}^{n} \gamma_{ll} \tag{2.23}$$

$$\text{Sujeito a:} \quad \sum_{j=1}^{i} \alpha_{ji} = 1 \qquad i = 1, ..., n \tag{2.24}$$

$$\alpha_{ji} = 0 \qquad j = 1, ..., n-1, i > j | w_i \neq w_j \vee h_i + h_j > H \tag{2.25}$$

$$\sum_{k=1}^{j} \beta_{kj} = \alpha_{jj} \qquad j = 1, ..., n \tag{2.26}$$

$$\sum_{i=j}^{n} h_i \alpha_{ji} \leq \sum_{k=1}^{j} h_k \beta_{kj} \qquad j = 1, ..., n-1 \tag{2.27}$$

$$\sum_{j=k}^{n} w_j \beta_{kj} \leq W \beta_{kk} \qquad k = 1, ..., n-1 \tag{2.28}$$

$$\sum_{l=1}^{k} \gamma_{lk} \leq \beta_{kk} \qquad k = 1, ..., n \tag{2.29}$$

$$\sum_{k=l}^{n} h_k \gamma_{lk} \leq H\gamma_{ll} \qquad l = 1, ..., n-1 \tag{2.30}$$

$$\alpha_{ji} \in \{0, 1\} \qquad j = 1, ..., n; i = j, ..., n \tag{2.31}$$

$$\beta_{kj} \in \{0, 1\} \qquad k = 1, ..., n; j = k, ..., n \tag{2.32}$$

$$\gamma_{lk} \in \{0, 1\} \qquad l = 1, ..., n; k = l, ..., n \tag{2.33}$$

Através de (2.23) minimiza-se o número de placas usadas. As restrições (2.24), (2.26) e (2.29) garantem que cada item, pilha e tira só podem ser empacotados exatamente uma vez (respetivamente). Em (2.25) é garantido que os itens na mesma pilha têm igual largura e que a altura total da placa não pode ser excedida e em (2.27) acautela-se que a altura das pilhas numa tira é inferior à altura do primeiro item na respetiva tira. A garantia de que as dimensões da placa não são ultrapassadas é assegurada pelas equações (2.28) e (2.30).

Tipicamente, a relaxação linear destes modelos de "atribuição" fornecem limites inferiores que são piores do que a relaxação linear dos modelos baseados em geração de colunas.

2.4.3 Heurísticas

Os métodos heurísticos são a abordagem mais utilizada nos modelos orientados aos itens para os problemas de corte a duas dimensões. Na literatura são descritas várias heurísticas construtivas para os problemas de empacotamento, isto é, em que os itens são considerados individualmente com procura igual a um.

Em Chung et al. (1982) é proposta uma heurística construtiva denominada de *Hybrid First Fit*(*HFF*). O algoritmo apresentado é dividido em duas fases. Em primeiro lugar os itens são ordenados por ordem decrescente de alturas e

depois empacotados sequencialmente em tiras segundo a regra "*first fit*" (o item é empacotado na primeira tira que o pode acondicionar). Na segunda fase, as tiras obtidas no primeiro passo são colocados nas placas segundo novamente a regra "*first fit*".

Extensões e modificações à heurística *HFF* foram propostas por Berkey & Wang (1987). A heurística *Finite Best Strip* (*FBS*) é muito similar à *HFF*, a grande diferença residindo na regra de alocação dos itens à tira e desta à placa, sendo neste caso a regra "*best fit*" (o item é alocado à tira com espaço residual horizontal mínimo). Propõem também duas heurísticas ligeiramente diferentes, em que os itens são diretamente alocados às placas. Na heurística *Finite First Fit* (*FFF*) os itens são ordenados por ordem decrescente de alturas sendo posteriormente empacotados sequencialmente nas placas. Um item é empacotado na primeira tira em que cabe, e se não couber em nenhuma tira de nenhuma placa existente, é iniciada uma nova placa. Na heurística *Finite Next Fit* (*FNF*) é apenas considerada uma placa de cada vez. Se um item não cabe em nenhuma tira da placa atual, esta placa é encerrada e uma nova placa é considerada. Deste modo, um item é colocado na tira atual da placa atual, se couber, caso contrário tenta-se construir uma nova tira na placa corrente, e, se tal também não for possível, uma nova placa é criada.

Os padrões obtidos segundo estas heurísticas são do tipo dois-estágios não-exato. Em Lodi et al. (1998) as heurísticas *FFF* e *FBS* foram adaptadas de modo a permitirem a rotação de 90^0 dos itens. Começam por ordenar os itens por ordem decrescente da sua aresta (largura ou altura) mais pequena e sempre que um item inicia uma nova tira ou placa, este é empacotado horizontalmente, isto é com a sua maior aresta paralela à parte inferior da tira ou da placa. Se for considerada uma tira já existente e se ambas as orientações forem admissíveis, escolhe-se a orientação vertical (aresta com menor valor paralelo à parte inferior

da tira ou da placa), de modo a favorecer outros empacotamentos na mesma tira. No mesmo trabalho, Lodi et al. (1998) propõem a heurística *Floor-Ceiling* (*FC*) cujo algoritmo é semelhante à heurística *FBS* adaptada para a rotação dos itens, a principal diferença consistindo na forma como os itens são empacotados nas tiras. Nesta heurística os itens além de poderem ser colocados na base da tira podem também ser posicionados na parte superior da tira (linha horizontal definida pela altura do item mais alto na tira). A heurística *FC* permite a rotação dos itens e além disso foi desenvolvida de modo a criar padrões de corte do tipo guilhotinado.

Lodi et al. (1999) propõem uma tipologia para os problemas de empacotamento considerando restrições relacionadas com a permissão de rotação dos itens e dos tipos de cortes, guilhotinados ou não. Da combinação destas restrições definem quatro problemas:

- 2BP|O|G - a rotação de 90^0 dos itens não é permitida (O) e exige-se que os cortes sejam guilhotinados (G);
- 2BP|R|G - os itens podem ser rodados 90^0 (R) e os cortes devem ser guilhotinados (G);
- 2BP|O|F - a rotação de 90^0 dos itens não é permitida (O) e os cortes podem ser não-guilhotinados (F);
- 2BP|R|F - os itens podem ser rodados 90^0 (R) e os cortes podem ser não-guilhotinados (F).

Apresentam heurísticas construtivas para as quatro versões do problema. Para o problema 2BP|O|G apresentam a heurística *Knapsack Packing* (*KP*), que consiste na resolução de um conjunto de problemas de mochila em dois passos. No

primeiro passo é obtido um conjunto de tiras, sendo depois resolvido um problema de empacotamento a uma dimensão. A tira é construída selecionando o item mais alto ainda não empacotado e resolvendo um problema de mochila, de modo a decidir quais os itens por empacotar que devem pertencer à tira. No problema de mochila, os benefícios são as áreas dos itens, os pesos são as larguras dos itens e a capacidade é limitada pela largura disponível.

O problema 2BP|R|G é resolvido através de uma adaptação na heurística (KP) de modo a permitir a rotação de 90^0 dos itens. Nesta adaptação os itens são ordenados por ordem decrescente da sua aresta (largura ou altura) mais pequena, tal como nas outras heurísticas para o problema em que é permitida a rotação dos itens.

Para o problema em que a orientação dos itens é fixa e os cortes são não guilhotinados 2BP|O|F, apresentam a heurística *Alternate Directions* (AD). Como o nome da heurística indica o algoritmo procura colocar os itens nas placas considerando direções alternadas, isto é, da esquerda para a direita e da direita para a esquerda. O algoritmo começa por ordenar os itens por ordem decrescente de altura e por calcular um limite inferior L para a solução ótima, depois são iniciadas L placas empacotando na base da primeira placa um subconjunto de itens usando a regra "*best fit*". Os restantes itens são empacotados em bandas de acordo com a direção atual associada à placa (esquerda para a direita ou direita para a esquerda). Para cada placa são explorados todos os itens ainda não alocados, tentando-se o empacotamento na direção corrente, se tal não for possível muda-se a direção e se nenhum item poder ser empacotado em nenhuma direção tenta-se a placa seguinte ou inicia-se uma placa vazia.

Finalmente, para o problema 2BP|R|F propõem a heurística *Touching Perimeter* (TP) que começa por ordenar os itens pelo valor decrescente das suas áreas e por calcular um limite inferior. O primeiro item colocado numa placa é sem-

pre posicionado no canto inferior esquerdo e cada item subsequente é colocado numa posição normal, segundo Christofides & Whitlock (1977), a base do item a empacotar deve tocar na base da placa ou na aresta superior de outro item e a sua aresta esquerda deve tocar na aresta esquerda da placa ou na aresta direita de outro item. A seleção da placa ou da posição de empacotamento é realizada tendo em conta o valor do parâmetro definido pela percentagem do perímetro do item que toca na placa ou num item já empacotado. Para cada posição candidata o valor do parâmetro é calculado duas vezes, um para cada orientação possível do item e é selecionada a que tiver maior valor.

No trabalho de Lodi et al. (1999) também é proposta uma pesquisa tabu que se adapta aos diferentes problemas (2BP|O|G, 2BP|R|G, 2BP|O|F, 2BP|R|F) mudando a heurística utilizada para pesquisar a vizinhança. A meta-heurística é formada por dois níveis conceptuais, um nível superior onde é selecionada a placa alvo a esvaziar e um nível inferior no qual se tenta empacotar os itens da placa alvo em subconjuntos de placas, é neste nível que é definido o tipo de problema abordado.

Em Lodi, Martello & Monaci (2002) e Lodi, Martello & Vigo (2002) é apresentada uma revisão dos métodos exatos, heurísticas, meta-heurísticas e limites inferiores na literatura para os problemas de empacotamento a duas dimensões.

A heurística construtiva *HBP* é proposta por Boschetti & Mingozzi (2003*a*) para o problema não-guilhotinado em que a rotação dos itens pode ou não ser permitida. Neste trabalho e em Boschetti & Mingozzi (2003*b*) são propostos limites inferiores e superiores para o mesmo problema. Na heurística *HBP* os itens são ordenados segundo um determinado critério e posteriormente são empacotados considerando-se uma placa de cada vez. Quando for impossível alocar itens à placa atual, esta é fechada e uma placa nova é aberta. Este procedimento para quando forem empacotados todos os itens, o processo é repetido alterando-se os

critérios de ordenação dos itens. A heurística para quando o número total de placas utilizadas é igual a um limite inferior ou é atingido o limite de iterações.

Um conjunto de abordagens heurísticas é proposto por Alvelos et al. (2009) para os problemas de empacotamento, considerando os problemas de dois e de três-estágios exatos e não-exatos. Utilizam heurísticas gulosas, de pesquisa local e descida em vizinhanças variáveis. As heurísticas gulosas baseiam-se no empacotamento sequencial dos itens, mantendo em memória os espaços livres que poderão receber os itens que ocorrem mais tarde na sequência. A localização na qual um item é empacotado é escolhida de acordo com um determinado critério. As várias heurísticas gulosas são obtidas através de modificações nos critérios de definição das sequências dos itens e nos critérios de empacotamento nos espaços disponíveis. Apresentam também três pesquisas locais, recorrendo à sequência de itens como representação de uma solução. Por último, a meta-heurística descida em vizinhanças variáveis explora as três estruturas de vizinhança utilizadas na pesquisa local. Neste trabalho, consideram também outras extensões ao problema, considerando um conjunto de restrições que advêm de duas indústrias do mobiliário.

Uma extensão a este trabalho é proposta por Pinto et al. (2009), para o problema em que as placas grandes estão disponíveis em vários tamanhos e em número limitado e são também considerados diferentes objetivos, além da minimização da área das placas usadas, pretendem também a maximização do valor das placas que sobram.

Em Chan et al. (2009, 2011*a*,*b*), para os problemas de empacotamento de dois e três-estágios exatos e não-exatos, são construídas soluções iniciais através de heurísticas construtivas, que são melhoradas através da pesquisa local com estruturas de vizinhança estocásticas e da pesquisa *Variable Neighborhood Descent* (VND).

Recentemente em Charalambous & Fleszar (2011) uma heurística construtiva orientada para a placa é apresentada. É abordado o problema guilhotinado com duas variantes, uma em que a rotação dos itens é permitida e outra em que é considerada a orientação fixa dos itens. A heurística construtiva orientada à placa, constrói soluções incrementalmente, adicionando itens a combinações de itens anteriormente geradas.

Capítulo 3

Problema de corte de dois e três-estágios [1]

Neste capítulo, é proposto um modelo de programação inteira para os problemas de corte de dois e três-estágios exatos e não exatos. São também consideradas outras variantes do problema como a possibilidade de rotação dos itens, a minimização do comprimento dos cortes e a atribuição de valores às placas residuais. O modelo de programação inteira proposto pode ser visto como uma extensão ao modelo "*one-cut*" proposto por Dyckhoff (1981) para o problema de corte a uma dimensão. No modelo proposto, cada variável de decisão está associada ao corte de um item de uma placa ou de parte de uma placa, que resulte de cortes anteriores (placas residuais). Foram realizados testes computacionais em instâncias provenientes de duas empresas da indústria do mobiliário, assim como em instâncias utilizadas na literatura.

[1] O trabalho descrito neste capítulo foi publicado em Silva et al. (2010).

3.1 Introdução

Neste capítulo é proposto um método exato para o problema de corte bidimensional. O método baseia-se na definição de um modelo de programação inteira, com um número pseudo-polinomial de variáveis e restrições, que é otimizado diretamente por um *solver* genérico de programação inteira. Ao contrário do que é usual na literatura, o método proposto não se baseia em geração de colunas para obter soluções ótimas para os problemas de corte a duas dimensões de dois e de três-estágios.

O modelo é uma extensão do modelo "*one-cut*" para o problema de corte a uma dimensão proposto por Dyckhoff (1981). O método baseia-se num modelo programação inteira que pode ser resolvido por um *solver* geral de programação inteira, tirando vantagem da eficiência e robustez que os *solvers* de programação inteira (-mista) alcançaram nos últimos anos. Por exemplo, no software usado nesta tese para os testes computacionais - CPLEX 11.0 (ver (Cplex11.0 2007)) - várias classes de desigualdades válidas e heurísticas são implementadas para melhorar os limites (inferiores e superiores) durante a pesquisa na árvore de partição e avaliação.

O modelo herda a possibilidade de modelar características já assinaladas por Dyckhoff, como múltiplos tipos de placas e como o valor das peças (placas) para uso futuro. Além disso, existe uma outra característica do modelo proposto que também é bastante relevante na prática, a modelação do comprimento (ou tempo) dos cortes necessários para obter todos os itens. Esta particularidade assim como uma extensão para a rotação dos itens, serão abordados após a apresentação do modelo.

Na próxima secção é introduzido o modelo de programação inteira para o problema de corte bidimensional para os problemas de dois e três-estágios exatos

e não-exatos. São também apresentados limites superiores para o número de restrições e para o número de variáveis de decisão do modelo. Uma vez que o modelo tem por base a enumeração de todos os cortes e placas residuais possíveis, é descrito um algoritmo para o cálculo dos mesmos. A secção 3.2 termina com um pequeno exemplo. Na secção 3.3 são discutidas e propostas extensões ao modelo desenvolvido. Os resultados computacionais são apresentados na secção 3.4. Os testes computacionais foram realizados em instâncias reais com origem em duas empresas da indústria do mobiliário e em instâncias da literatura. Os resultados fornecidos pelo modelo de programação inteira proposto são comparados com os dois modelos de afetação e com os dois modelos de geração de colunas da literatura. Na última secção são efetuadas as principais conclusões deste capítulo.

3.2 Modelo de programação inteira

3.2.1 Descrição

O modelo proposto baseia-se na consideração explícita de como os itens procurados podem ser obtidos a partir das placas. O conceito principal do modelo apresentado é o *corte*, que consiste em utilizar uma placa e através dela obter um item (efetuando um ou dois cortes em guilhotina). De um modo geral, de um corte resulta um item e duas placas que poderão ainda ser cortadas. As placas que resultam de um corte denominam-se *placas residuais*. De modo a ilustrar estes conceitos, apresenta-se um pequeno exemplo.

Exemplo 3.1 Considere o problema de corte bidimensional de dois-estágios não-exato com placas iniciais com largura e altura 6 e dois tipos de itens, indexados por i, com larguras, alturas e procuras dadas na tabela 3.1.

Tabela 3.1: Dados do exemplo.

Tipo de item	Largura	Altura	Procura
1	4	3	5
2	2	2	5

Obter um item do tipo um de uma placa inicial é um exemplo de corte. Neste corte, uma tira de altura 3 é gerada através de um corte horizontal e um item do tipo um é depois obtido por um corte vertical na tira. Como resultado, além do item do tipo um, duas novas placas com os tamanhos $(6, 3)$ e $(2, 3)$ são obtidas, conforme ilustrado na figura 3.1, onde as linhas a negrito representam os cortes. Outro exemplo de corte é a obtenção de um item do tipo dois a partir da placa residual com dimensões $(6, 3)$ que é ilustrado na figura 3.2. Como resultado, além do item do tipo dois, duas novas placas com tamanhos $(6, 1)$ e $(4, 2)$ são criadas. A placa $(6, 1)$ é considerada desperdício, pois tem uma altura menor que a menor altura dos itens.

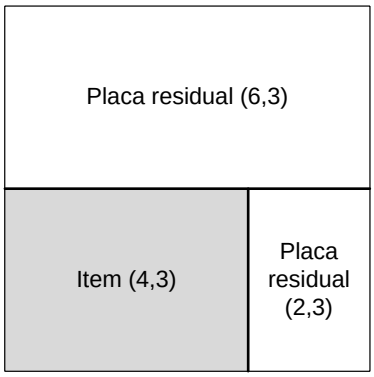


Figura 3.1: Corte de um item do tipo um de uma placa do tipo inicial.

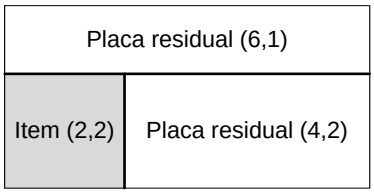


Figura 3.2: Corte de um item do tipo dois de uma placa residual.

O modelo proposto baseia-se na enumeração de todos os cortes e de todas as placas residuais. Placas do mesmo tipo têm a mesma largura, altura e pertencem ao mesmo estágio. O estágio é relevante aquando da definição do tipo de placa, porque, como ilustrado na figura 3.3 e na figura 3.4, um corte produz diferentes placas de acordo com o estágio em que é realizado.

Figura 3.3: Corte no primeiro ou terceiro estágios.

Há um conjunto de regras simples que permite definir os cortes possíveis, calculando a largura, a altura e o estágio das placas residuais consequentes. Na subsecção 3.2.3 são analisados os detalhes da geração dos cortes e das placas residuais. Nesta subsecção assume-se que todos os cortes e os tipos de placas residuais possíveis são conhecidos.

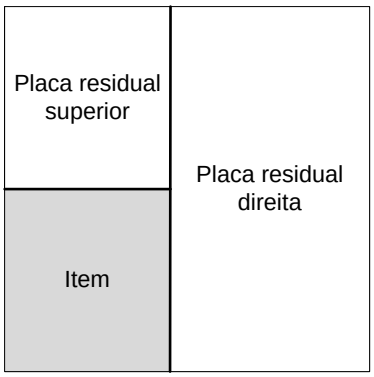


Figura 3.4: Corte no segundo estágio.

A ideia principal do modelo de programação inteira proposto é associar uma variável de decisão a cada corte, mais precisamente ao número de vezes que o corte é efetuado. As restrições garantem que são cortados todos os itens pedidos e que os cortes apenas podem ser realizados em placas existentes.

O modelo é agora apresentado formalmente. O número de tipos de itens é definido como n e o número de tipos de placas residuais é definido como m. Os tipos de placas são indexadas por j, $j = 0, ..., m$, $j = 0$ representando as placas iniciais e $j = 1, ..., m$, caraterizando os tipos de placas residuais. A procura do item do tipo i, $i = 1, ..., n$, é designada por d_i .

A variável de decisão $x_{ij}, i = 1, ..., n, j = 0, ..., m$, é definida como o número de vezes que o item do tipo i é cortado da placa do tipo j. O parâmetro $a_{ijk}, i = 1, ..., n, j = 0, ..., m, k = 1, ..., m$, é igual a 1 se a placa do tipo k resulta de cortar o item do tipo i da placa do tipo j, e igual a 0 caso contrário. Utilizando a notação introduzida, segue-se o modelo.

$$\text{Minimizar} \quad \sum_{i=1}^{n} x_{i0} \tag{3.1}$$

$$\text{Sujeito a:} \quad \sum_{j=1}^{m} x_{ij} \geq d_i, \qquad i = 1, ..., n, \tag{3.2}$$

$$\sum_{j=0}^{m} \sum_{i=1}^{n} a_{ijk} x_{ij} \geq \sum_{i=1}^{n} x_{ik} \qquad k = 1, ..., m, \tag{3.3}$$

$$x_{ij} \geq 0 \text{ e inteiro}, \qquad i = 1, ..., n, j = 0, ..., m. \tag{3.4}$$

A função objetivo (3.1) minimiza o número de cortes efetuados numa placa do tipo inicial. O conjunto de restrições (3.2) garante que a procura de cada tipo de item é satisfeita. No modelo apresentado, é permitida a produção em excesso (exceder as procuras). Se as restrições de maior ou igual (3.2) forem substituídas por igualdades, o modelo não permite produção excedentária, isto é, serão cortadas as quantidades exatas de cada tipo de item. As restrições (3.3) acautelam que, para cada tipo de placa residual, o número de cortes que produzem essas placas residuais deve ser maior ou igual ao número de cortes efetuados nessas placas, isto é, só podem ser executados cortes em placas residuais existentes.

O tamanho do modelo proposto depende da dimensão dos itens, das placas e do número de tipos de itens. De uma forma geral, instâncias com itens pequenos (quando comparados com a dimensão das placas) são mais difíceis de resolver (de uma forma exata) do que instâncias com itens maiores, devido à estrutura combinatória do problema (com itens pequenos, são possíveis mais disposições na placa). No método proposto, ter itens pequenos traduz-se num maior número de cortes e de placas residuais possíveis, o que contribui para um modelo de programação inteira maior. O tamanho do modelo depende também do número de tipos de itens. Comparando instâncias com o mesmo número de itens, o modelo proposto é mais apropriado para instâncias com poucos tipos de itens (com procuras grandes) do que o oposto, pois as procuras não afetam o tamanho do modelo. Na subsecção seguinte são dados limites superiores para as dimensões do modelo de programação inteira.

3.2.2 Tamanho

Nesta subsecção, são apresentados três limites superiores para o número de placas residuais para o problema de dois-estágios e três limites superiores para o número de placas residuais para o problema de três-estágios. Estes limites superiores podem facilmente ser utilizados para obter limites superiores para o número de cortes. Tendo em conta que no modelo de programação inteira as restrições e as variáveis estão relacionadas com o número de tipos de placas residuais e com o número de cortes, respetivamente, os limites superiores mostram que o modelo tem tamanho pseudo-polinomial.

Será utilizada a seguinte notação. Como anteriormente, n é número de tipos de itens, W e H são a largura e a altura da placa inicial e w' e h' são a largura mínima e a altura mínima dos tipos de itens, respetivamente. Um limite superior para o número de tipos de placas residuais para o problema de dois-estágios baseado na estrutura combinatória do problema é apresentado na Proposição 3.1.

Proposição 3.1 *Para o problema de dois-estágios não-exato, um limite superior para o número de tipos de placas residuais é dado por*

$$UB2^C = n + n^2 + ... + n^{\left\lfloor \frac{H}{h'} \right\rfloor} + n + n^2 + ... + n^{\left\lfloor \frac{W}{w'} \right\rfloor}.$$

Prova. Existem dois tipos de placas residuais: as que estão à direita do item a cortar e as acima do item (ver figuras 3.3 e 3.4). Serão designadas por placa residual superior e placa residual direita, respetivamente. Primeiro, será apresentado um limite superior para as placas residuais superiores.

Considerando o problema de dois-estágios não-exato, as placas residuais superiores resultam apenas de cortes em placas iniciais (figura 3.3) ou de outras placas residuais superiores. Existe um total de n cortes associados à placa inicial, cada um produzindo uma placa residual superior. Para cada placa residual superior, um limite superior para o número de cortes é n, cada um produzindo

n placas residuais superiores. Um limite superior para o número de vezes que este processo pode ser repetido é $\left\lfloor \frac{H}{h'} \right\rfloor$. Deste modo, um limite superior para o número de placas residuais superiores é $n + n^2 + ... + n^{\left\lfloor \frac{H}{h'} \right\rfloor}$.

Utilizando o mesmo raciocínio, é obtido um limite superior para as placas residuais à direita. Note-se que, no pior caso, existem n alturas diferentes para as placas residuais à direita (uma para cada tipo de item). A partir de qualquer placa (inicial ou residual), o número de cortes é n e cada um produz uma placa residual direita com a mesma altura. Um limite superior para o número de placas residuais à direita é $n + n^2 + ... + n^{\left\lfloor \frac{W}{w'} \right\rfloor}$.

Na Proposição 3.2, é dado outro limite superior para o número de tipos de placas residuais baseado em discretização.

Proposição 3.2 *Para o problema de dois-estágios não-exato, assumindo que* W, H, w' *e* h' *são inteiros não negativos, um limite superior para o número de tipos de placas residuais é dado por*

$$UB2^d = (H - h') + (W - w')n.$$

Prova. O limite superior para as placas residuais superiores $(H - h')$ é obtido diretamente por discretização. O limite superior para o número de placas residuais à direita é também obtido diretamente, sabendo que para cada uma das n alturas possíveis há $(W - w')$ larguras possíveis, por discretização.

Tendo em conta que nenhum limite superior para o número de tipos de placas residuais domina o outro, o melhor limite superior é dado na Proposição 3.3.

Proposição 3.3 *Para o problema de dois-estágios, um limite superior do número de tipos de placas residuais é dado por*

$$UB2^b = Min\{UB2^c, UB2^d\}.$$

As três proposições seguintes são para o problema de três-estágios. A altura máxima de um tipo de item é denotada por h^*.

Proposição 3.4 *Para o problema de três-estágios não-exato, um limite superior para o número de tipos de placas residuais é dado por*

$$UB3^c = n + n^2 + ... + n^{\left\lfloor \frac{H}{h'} \right\rfloor} + n + n^2 + ... + n^{\left\lfloor \frac{W}{w'} \right\rfloor} + n + n^2 + ... + n^{\left\lfloor \frac{h^* - h'}{h'} \right\rfloor}.$$

Prova. Todas as placas residuais consideradas no problema de dois-estágios existem para o problema de três-estágios não-exato. Para o problema de três-estágios, são consideradas placas residuais superiores adicionais, que resultam de cortes nas placas residuais à direita e em placas residuais superiores resultantes de cortes das placas residuais à direita. O número destas placas residuais superiores é dado por $n + n^2 + ... + n^{\left\lfloor \frac{h^* - h'}{h'} \right\rfloor}$.

Proposição 3.5 *Para o problema de três-estágios não-exato, assumindo que, W, H, w', h' e h^* são inteiros não negativos, um limite superior para o número de tipos de placas residuais é dado por*

$$UB3^d = (H - h') + n(W - w')(h^* - h').$$

Prova. O limite superior para o número de placas residuais superiores $(H - h')$ é obtido diretamente por discretização. Para cada uma das n alturas possíveis, por discretização, há $(W - w')$ larguras possíveis de placas residuais à direita (segundo estágio). Para cada uma destas larguras, no pior caso, existem $h^* - h'$ alturas diferentes.

Proposição 3.6 *Para o problema de três-estágios não-exato, um limite superior para o número de tipos de placas residuais é dado por*

$$UB3^b = Min\{UB3^c, UB3^d\}.$$

Como o número de restrições do modelo é n mais o número de tipos de placas residuais, um limite superior para o número de restrições do modelo para os problemas de dois-estágios é $n + UB2^b$ e, para o problema de três-estágios, é $n + UB3^b$.

Considerando, no pior caso, que cada tipo de placa pode ser usada para cortar qualquer tipo de item, um limite superior para o número de variáveis de decisão para o problema de dois-estágios é $n(1+UB2^b)$, e para o problema de três-estágios é $n(1 + UB3^b)$.

3.2.3 Geração dos cortes e das placas residuais

O modelo de programação inteira baseia-se na enumeração de todos os cortes e placas residuais possíveis. O algoritmo para a geração de todos os cortes e placas residuais possíveis é dado no Algoritmo 1. A lista dos tipos de placas residuais é denominada por R, e a lista de todos os cortes possíveis é designada por C. Em ambas as listas, não é permitida repetição. Um conjunto de regras para definir a largura, a altura e o estágio das placas residuais resultantes de um corte são dados na tabela 3.2 (problemas de dois-estágios) e na tabela 3.3 (três-estágios).

Ao decidir se um corte deve ser considerado (ver as condições do **se** dentro do ciclo **para todos** no Algoritmo 1), se considerarmos o problema de dois-estágios exato, devem ser considerados apenas cortes de itens com a mesma altura que a placa residual no estágio dois. Se o problema a considerar for o de três-estágios, apenas devem ser considerados cortes de itens tendo a mesma largura que a placa residual no estágio três.

O algoritmo para a enumeração dos cortes e dos tipos de placas residuais é usado para construir o modelo de programação inteira. Como cada coluna do modelo corresponde a um corte, uma representação dos elementos não-zeros da matriz é usada. Cada coluna tem, no máximo, quatro elementos não-zero: 1 na

Algoritmo 1: Algoritmo para gerar cortes e placas residuais.

início

Inicializar R com o conjunto vazio;

Inicializar C com n cortes correspondendo ao corte de um tipo de item da placa inicial;

Marcar todos os cortes C como não analisados;

enquanto *não forem analisados todos os cortes de* C **faça**

Seja c um corte não analisado de C associado com o tipo de item i e um tipo de placa;

Marcar c como analisado;

Adicionar os dois tipos de placas (superior e à direita), definidos pelas larguras, alturas e estágios dados pelas regras da tabela 3.2 (problema de dois-estágios) ou tabela 3.3 (problema de três-estágios) a R, se estas ainda lá não estiverem;

para todo *itens do tipo* i **faça**

se *o item do tipo* i *pode ser cortado de uma placa do tipo superior* **então**

Adicione o corte associado à placa superior e ao item do tipo i a C e marque-o como não analisado

fim

se *o item do tipo* i *pode ser cortado de uma placa do tipo à direita* **então**

Adicione o corte associado à placa do tipo à direita e ao item do tipo i a C e marque-o como não analisado

fim

fim

fim

fim

Tabela 3.2: Tipos de placas resultantes de cortes em diferentes estágios para o problema de dois-estágios.

Cortes					Tipos de placas resultantes					
Tipo de placa			Tipo de item		Superior			Direita		
Largura	Altura	Estágio	Largura	Altura	Largura	Altura	Estágio	Largura	Altura	Estágio
W_r	H_r	1	w_i	h_i	W_r	$H_r - h_i$	1	$W_r - w_i$	h_i	2
W_r	H_r	2	w_i	h_i	W_i	$H_r - h_i$	desperdício	$W_r - w_i$	H_r	2

Tabela 3.3: Tipos de placas resultantes de cortes em diferentes estágios para o problema de três-estágios.

Cortes					Tipos de placas resultantes					
Tipo de placa			Tipo de item		Superior			Direita		
Largura	Altura	Estágio	Largura	Altura	Largura	Altura	Estágio	Largura	Altura	Estágio
W_r	H_r	1	w_i	h_i	W_r	$H_r - h_i$	1	$W_r - w_i$	h_i	2
W_r	H_r	2	w_i	h_i	W_i	$H_r - h_i$	3	$W_r - w_i$	H_r	2
W_r	H_r	3	w_i	h_i	W_i	$H_r - h_i$	3	$W_r - w_i$	h_i	desperdício

linha da procura do tipo de item associado ao corte, um -1 na linha da placa residual associada ao corte, e dois 1 nas linhas das placas residuais resultantes. Quando um corte envolve um tipo de placa residual, cuja linha não foi previamente gerada, a linha é criada e adicionada à matriz das restrições, usando um formato compacto. Cada corte é considerado apenas uma vez. Para cada corte são verificadas todas as placas residuais já geradas, de modo a não permitir repetições.

O ciclo **enquanto** depende do número de cortes e em cada iteração é necessário verificar se a placa residual gerada está contida na lista R. Usando estruturas de dados básicas para implementar o algoritmo, o pior caso de complexidade computacional do algoritmo é $O(|C|\,|R|)$. Usando os limites superiores do número de tipos de placas residuais e cortes, apresentados na subsecção anterior, o pior

caso de complexidade computacional do algoritmo é $O(n^{1+max\{\lfloor \frac{H}{h'} \rfloor, \lfloor \frac{W}{w'} \rfloor\}})$ para os problemas de dois e três-estágios. Na prática, tal como é apresentado na secção 3.4, o tempo de construção do modelo é sempre muito inferior ao tempo computacional despendido na otimização do modelo.

Um procedimento para reduzir a simetria do modelo (várias soluções equivalentes correspondentes a diferentes valores das variáveis de decisão), que é frequentemente utilizado nos modelos para os problemas de corte, é apresentado para o problema de dois-estágios não-exato e para o problema de três-estágios. Para o problema de dois-estágios não-exato, este procedimento pode ser visto como uma ordenação dos itens nas tiras por ordem decrescente de altura. Quando um corte é executado numa placa residual no estágio dois, a placa residual à direita resultante terá a altura do item e não a altura da placa residual. Desta forma, todos os cortes com base na placa residual à direita (que também é uma placa residual no estágio dois) serão associados a itens com altura inferior ou igual à altura do item.

Uma justificação semelhante pode ser aplicada ao problema de três-estágios não-exato. Neste caso, os itens nas pilhas são ordenados por ordem decrescente de largura. Quando um corte é efetuado numa placa residual no estágio três, a placa residual superior resultante terá a largura do item e não a largura da placa residual que lhe deu origem. Estes procedimentos apenas afetam a ordem dos itens na tira (problema de dois-estágios) ou na pilha (problema de três-estágios) e não o valor da solução.

3.2.4 Exemplo

A matriz das restrições do modelo de programação inteira para o exemplo 3.1 apresentada no início da subsecção 3.2.1 é apresentada na tabela 3.4. A largura, a altura e o estágio dos tipos de placas residuais são dados na tabela 3.5. Cada

Tabela 3.4: Matriz das restrições para a instância do exemplo.

Tipo de item i		1	2	1	2	2	1	2	2	2	2
Tipo de placa j		0	0	1	1	2	3	3	4	5	6
Tipo de item	1	1		1			1				
	2		1		1	1		1	1	1	1
Tipo de placa	1	1		-1	-1						
	2	1		1		-1	1				
	3		1				-1	-1			
	4		1		1			1	-1	1	
	5							1		-1	
	6								1		-1

coluna da matriz da tabela 3.4 está associada a uma variável de decisão que é constituída por:

- um 1 na primeira linha da matriz se o corte correspondente é para o item do tipo um, 1 na segunda linha se o corte correspondente é para um item do tipo dois;
- um -1 na linha associada à placa residual onde o corte é executado (se o corte é na placa inicial, a coluna não tem -1);
- um valor 1 em cada linha associada com o tipo de placa residual resultante (no máximo duas linhas).

Por exemplo, a terceira coluna (associada à variável x_{11}) corresponde ao corte do item do tipo um da placa do tipo um (largura 6, altura 3, primeiro estágio) e resultando numa placa do tipo dois (largura 2, altura 3, estágio 2).

Uma solução ótima é $x_{10} = 3, x_{11} = 2, x_{21} = 1, x_{22} = 2, x_{24} = 1, x_{26} = 1$, com todas as outras variáveis de decisão com valor igual a 0. Relembra-se que x_{ij} é o número de vezes que o item do tipo i é cortado da placa do tipo j.

Tabela 3.5: Tamanhos e estágios dos tipos de placas.

Tipo de placa	Largura	Altura	Estágio
0	6	6	1
1	6	3	1
2	2	3	2
3	6	4	1
4	4	2	2
5	6	2	1
6	2	2	2

A sequência de cortes começa numa placa inicial e o processo continua analisando os cortes nas placas residuais resultantes. A interpretação desta solução deve ser feita da seguinte forma. Como $x_{10} = 3$ e $x_{20} = 0$, são usadas três placas iniciais, e em cada uma delas é cortado um item do tipo um. Tal como é ilustrado na matriz das restrições e na figura 3.5 (onde os números são os índices dos tipos de placas residuais), estes cortes produziram três placas residuais do tipo um e três placas residuais do tipo 2.

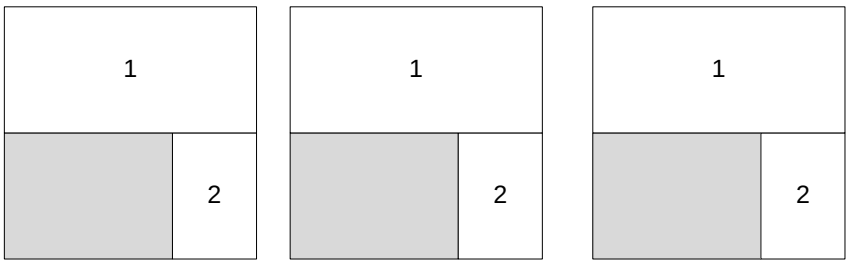


Figura 3.5: Cortes nas placas iniciais.

Uma vez que, $x_{11} = 2$ e $x_{21} = 1$, dois itens do tipo um são cortados da placa residual do tipo 1 e um item do tipo dois é cortado da terceira placa residual do tipo um (Ver figura 3.6).

Figura 3.6: Cortes nas placas residuais do tipo um.

Como $x_{22} = 2$, são cortados dois itens do tipo dois de duas das cinco placas residuais do tipo dois, como ilustrado na figura 3.7.

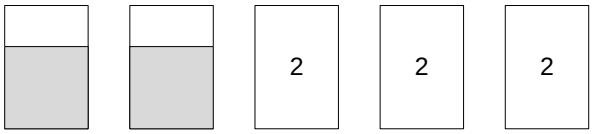


Figura 3.7: Cortes nas placas residuais do tipo dois.

Tendo em conta que $x_{24} = 1$, um item do tipo dois é cortado da placa residual do tipo quatro, como pode ser observado na figura 3.8.

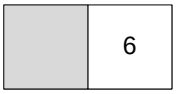


Figura 3.8: Corte na placa residual do tipo quatro.

Por último, como $x_{26} = 1$, a placa residual do tipo seis é usada como um item do tipo dois. Três placas residuais do tipo dois não serão cortadas. Uma

solução constituída por cortes pode ser também representada como padrões e vice-versa. Para o exemplo, uma possível representação dos padrões para a solução apresentada anteriormente é dada na figura 3.9.

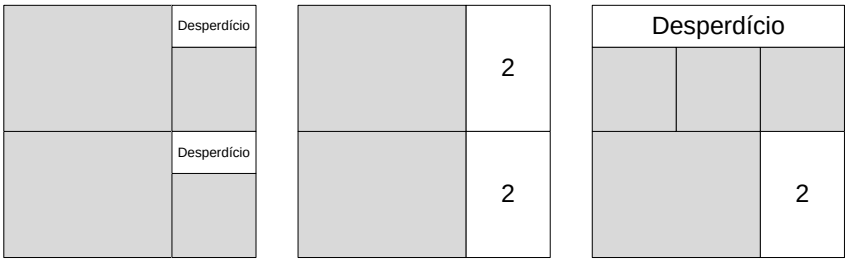


Figura 3.9: Solução representada na forma de padrões de corte.

3.3 Extensões

3.3.1 Rotação

Em alguns problemas de corte bidimensional, é permitida a rotação de 90 graus dos itens. É fácil lidar com esta extensão no modelo, o corte é agora definido por um tipo de placa, um tipo de item e a orientação do item. As variáveis de decisão passam a ser x_{irj}, significando o número de vezes que o item i com a orientação r ($r = 0$ significa que o item não é rodado e $r = 1$ significa que o item é rodado) é cortado da placa do tipo j. A geração dos cortes e das placas residuais requer apenas pequenas modificações, tendo em conta que cada item pode ser cortado de duas formas diferentes de uma placa.

O modelo é o seguinte:

$$\text{Minimizar} \quad \sum_{i=1}^{n}\sum_{r=0}^{1} x_{ir0} \tag{3.5}$$

$$\text{Sujeito a:} \quad \sum_{j=0}^{m}\sum_{r=0}^{1} x_{irj} \geq d_i, \qquad i = 1, ..., n, \tag{3.6}$$

$$\sum_{j=0}^{m}\sum_{i=1}^{n}\sum_{r=0}^{1} a_{irjk} x_{irj} \geq \sum_{i=1}^{n}\sum_{r=0}^{1} x_{irk} \qquad k = 1, ..., m, \tag{3.7}$$

$$x_{irj} \geq 0 \text{ e inteiro}, \qquad i = 1, ..., n; r = 0, 1; j = 0, ..., m. \tag{3.8}$$

3.3.2 Múltiplos tipos de placas

A extensão para o problema de corte a duas dimensões com placas de vários tamanhos é simples. A única diferença significativa é a inicialização da lista de cortes, ao gerar o modelo de programação inteira.

3.3.3 Comprimento dos cortes

Os objetivos usualmente considerados nos problemas de corte e empacotamento são a minimização do número de placas usadas e a minimização dos desperdícios (que será discutido na próxima subsecção). Habitualmente, em termos práticos, problemas relacionados com o comprimento dos cortes são também relevantes. Por exemplo, o comprimento total dos cortes está relacionado com o tempo despendido na execução da operação de corte, podendo ser uma característica importante para a gestão da produção.

Um exemplo de duas soluções com o mesmo número de placas, mas diferentes comprimentos totais dos cortes é dado na figura 3.10. Assumindo que a placa inicial tem altura 4 e largura 2, conclui-se facilmente que para a primeira solução é necessário um corte de comprimento 8, enquanto para a segunda solução é necessário apenas um corte de comprimento 7.

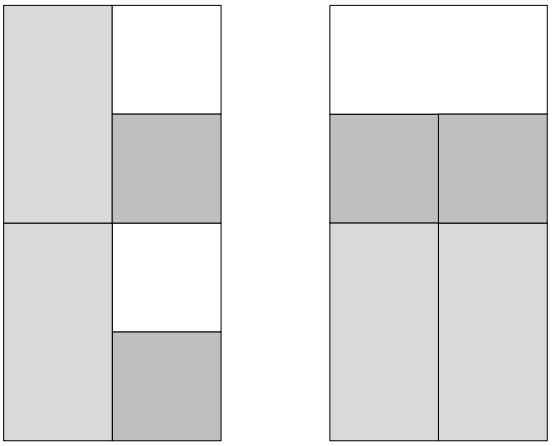


Figura 3.10: Duas soluções com uma placa, mas com diferentes número e comprimento dos cortes (espaços em branco são desperdício).

O modelo pode incorporar esta característica, uma vez que é fácil calcular o comprimento da operação de corte associado a um corte. Para um corte numa placa residual no estágio um ou três (ver figura 2.3) o comprimento dos cortes é $W_j + h_i$, onde W_j é a largura da placa residual, se as alturas e as larguras da placa residual e do item são diferentes. Se a altura é igual, então o comprimento do corte é h_i. Se a largura é a mesma, então o comprimento do corte é W_j. Condições e cálculos similares podem ser realizados para os cortes em placas residuais no estágio dois (ver figura 2.4).

Representando o comprimento dos cortes do corte ij (item i cortado da placa do tipo j) por l_{ij}, o comprimento total dos cortes é dado por:

$$\sum_{i=1}^{n} \sum_{j=0}^{m} l_{ij} x_{ij}.$$

Esta função (linear) pode ser usada como restrição, de forma a garantir que o comprimento dos cortes é inferior ou igual a um determinado valor, ou como função objetivo. No último caso, o número de placas usadas pode ser incluída no

modelo como uma restrição. Outro uso possível para a função apresentada é o de selecionar uma solução de entre as soluções (ótimas) alternativas com o mesmo número de placas. Neste caso, a função pode ser considerada como um termo adicional com um pequeno peso na função objetivo.

3.3.4 Minimização dos desperdícios

No final do processo de corte existem três tipos de peças: itens, placas de desperdício (aquelas em que não é possível cortar qualquer item) e placas residuais vazias (placas residuais em que nenhum item é cortado).Ver a figura 3.9 para uma ilustração destes três tipos de peças. Quando se minimiza o número de placas usadas, apenas são necessárias variáveis de decisão relacionadas com as placas residuais. Contudo, quando se minimiza os desperdícios, é necessário quantificar a área dos desperdícios e a área das placas residuais vazias. Desta forma, para a minimização dos desperdícios, são definidos dois novos conjuntos de variáveis de decisão: w_p é o número de placas desperdício do tipo $p, w_p \geq 0$ e inteiro, $p = 1, ..., q$, onde q é o número de tipos de placas desperdício; e $s_k, s_k \geq 0$, é o número de placas residuais do tipo k vazias, $k = 1, ..., m$. As últimas variáveis são as variáveis de excesso das restrições das placas residuais, que passam agora a ser restrições de igualdade:

$$s_k = \sum_{j=0}^{m} \sum_{i=1}^{n} a_{ijk} x_{ij} - \sum_{i=1}^{n} x_{ik}, \qquad k = 1, .., m.$$

Definindo a_p como a área de uma placa de desperdício do tipo $p, p = 1, ..., q$, e c_k como a área da placa residual do tipo k, $k = 1, ..., m$, respetivamente, a função objetivo para a minimização dos desperdícios é

$$Min \sum_{p=1}^{q} a_p w_p + \sum_{k=1}^{m} c_k s_k.$$

3.3.5 Placas residuais de sobra

No modelo, é simples tratar as peças que não são cortadas como stock, com um determinado valor, para usar no futuro (próximo período de planeamento).

Usando as variáveis de excesso introduzidas na subsecção anterior, pode atribuir-se um valor às placas residuais, passando estas a serem consideradas na função objetivo. As placas residuais que não possam ser usadas no futuro, particularmente por causa da sua área (largura ou altura) são consideradas desperdício, tendo valor zero. Valores proporcionais à área (por exemplo) podem ser dados às outras placas residuais. De um modo geral, pode associar-se qualquer valor a qualquer placa residual.

3.4 Resultados computacionais

3.4.1 Ambiente computacional e instâncias testadas

Com o objetivo de efetuar a avaliação do desempenho do modelo proposto, nesta secção apresentam-se os testes computacionais realizados com instâncias provenientes da indústria do mobiliário e com instâncias adaptadas da literatura.

Considerando a divisão da revisão bibliográfica em dois grandes grupos apresentada no capítulo anterior, métodos de solução orientados ao padrão e métodos de solução orientados ao item, o modelo proposto será comparado com dois modelos de geração de colunas e dois modelos de afetação.

Deste modo, os testes computacionais dividem-se em dois grupos. No primeiro grupo os resultados obtidos com o modelo proposto são comparados com os resultados dos modelos de afetação de Lodi et al. (2004) e de Puchinger & Raidl (2007) para os problemas de dois-estágios não-exato e três-estágios exato, respetivamente. No segundo grupo o modelo proposto é comparado com dois mo-

delos de geração de colunas, a abordagem clássica de Gilmore & Gomory (1965) e um novo modelo baseado em geração de colunas em que o subproblema é resolvido por um conjunto de problemas de corte a uma dimensão com a formulação de Dyckhoff (1981), esta comparação é realizada para os problemas em que os padrões são do tipo dois-estágios não-exato.

A construção e otimização do modelo proposto foi implementada em C++, utilizando a biblioteca de funções do *solver ILOG Cplex 11.*0 (*Ilog 2007*). O mesmo *solver* foi utilizado para otimizar os modelos de programação inteira de Lodi et al. (2004) para o problema de dois-estágios não-exato e de Puchinger & Raidl (2007) para o problema de três-estágios exato e para os modelos de geração de colunas. Em todos os casos foram utilizados os parâmetros por omissão do *Cplex*. Todos os testes computacionais foram efetuados num processador 1.87 *GHz Intel Core Duo* com 2GB de memória *RAM*.

As características dos quatro conjuntos de instâncias testadas podem ser consultadas da tabela 3.6 à tabela 3.9. Os conjuntos de instâncias reais foram fornecidas por duas empresas da indústria do mobiliário, um conjunto é constituído por 47 instâncias, enquanto o outro contém 121 instâncias. As instâncias da primeira empresa (A) são heterogéneas, isto é, algumas tem valores pequenos de procuras (poucas unidades) e outras têm grandes procuras (dezenas). As instâncias da segunda empresa (B) têm procuras grandes ou até mesmo muito grandes (dezenas ou centenas, ou até mesmo milhares).

Tabela 3.6: Instâncias da empresa *A*.

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
AP-9_1	30	770	25,7	2550	2100	145	80
AP-9_10	20	559	28,0	2550	2100	221	109
AP-9_11	27	507	18,8	2550	2100	171	141
AP-9_12	10	215	21,5	2550	2100	321	144
AP-9_13	21	450	21,4	2550	2100	304	141

Tabela 3.6 Instâncias da empresa *A (continuação da página anterior).*

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
AP-9_14	1	24	24,0	2550	2100	431	1001
AP-9_15	8	248	31,0	2550	2100	321	281
AP-9_2	3	44	14,7	2550	2100	466	80
AP-9_3	20	724	36,2	2550	2100	225	250
AP-9_4	3	44	14,7	2550	2100	481	952
AP-9_5	8	304	38,0	2550	2100	367	80
AP-9_6	31	809	26,1	2550	2100	301	210
AP-9_7	12	339	28,3	2550	2100	445	314
AP-9_8	27	744	27,6	2550	2100	301	109
AP-9_9	3	135	45,0	2550	2100	363	318
AP-9-3MM-4MM_1	2	24	12,0	2750	1220	463	420
AP-9-3MM-4MM_2	4	38	9,5	2750	1220	303	386
AP-9-3MM-4MM_3	2	17	8,5	2750	1220	463	386
AP-9-3MM-4MM_4	2	16	8,0	2750	1220	76	375
AP-9-3MM-4MM_5	8	138	17,3	2470	2080	303	210
AP-9-3MM-4MM_6	2	17	8,5	2750	1220	736	910
AP-9-3MM-4MM_7	5	58	11,6	2750	1220	706	286
AP-9-3MM-4MM_8	1	16	16,0	2750	1220	486	310
FA+AA-9_1	107	217	2,0	2550	2100	267	91
FA+AA-9_10	36	64	1,8	2550	2100	406	101
FA+AA-9_11	98	184	1,9	2550	2100	161	191
FA+AA-9_12	74	150	2,0	2550	2100	321	151
FA+AA-9_13	134	309	2,3	2550	2100	126	101
FA+AA-9_14	26	46	1,8	2550	2100	206	281
FA+AA-9_15	68	144	2,1	2550	2100	161	249
FA+AA-9_2	75	156	2,1	2550	2100	251	145
FA+AA-9_3	110	257	2,3	2550	2100	210	91
FA+AA-9_4	34	61	1,8	2550	2100	328	80
FA+AA-9_5	54	120	2,2	2550	2100	151	81
FA+AA-9_6	78	180	2,3	2550	2100	267	109
FA+AA-9_7	54	106	2,0	2550	2100	221	101
FA+AA-9_8	82	218	2,7	2550	2100	267	81
FA+AA-9_9	24	39	1,6	2550	2100	267	131
FA+AP-9-10MM_1	16	52	3,3	2550	2100	477	145
FA+AP-9-10MM_2	8	78	9,8	2550	2100	353	250
FA+AP-9-10MM_3	42	160	3,8	2550	2100	273	74
FA+AP-9-10MM_4	11	22	2,0	2550	2100	273	90
TRAS-BC_1	37	75	2,0	2470	2080	143	74
TRAS-BC_2	40	163	4,1	2750	1220	250	150
TRAS-BC_3	32	71	2,2	2750	1220	76	143
TRAS-BC_4	8	13	1,6	2750	1220	703	320
TRAS-BC_5	11	22	2,0	2750	1220	228	143

Tabela 3.7: Instâncias da empresa B.

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
03G18_1	9	757	84,1	4250	2120	324	360
03J18_2	13	1040	80	4250	2120	324	360
03X22_3	5	1342	268,4	4250	1860	396	396
10C30_4	7	123	17,6	4250	1860	880	360
10M30_5	4	118	29,5	4250	1860	460	330
10W30_6	3	42	14,0	4250	1860	1400	925
114M18_7	4	3000	750	2440	1860	464	331
114UU18_9	6	2545	424,2	2800	1860	438	540
12D16_10	21	2629	125,2	4250	1680	347	260
12F18_11	21	2838	135,1	4250	1860	273	287
12M18_12	31	6134	197,9	4250	1860	318	285
146O18_17	6	6570	1095,0	3760	1860	423	320
146OO18_18	13	5871	451,6	3760	1860	422	290
15A30_19	7	318	45,4	4250	2120	670	458
15B30_20	5	106	21,2	4250	2200	808	418
15G30_21	8	316	39,5	4250	2200	670	484
15J30_22	2	57	28,5	4250	2120	2069	484
15M30_23	3	147	49,0	4250	2120	670	484
15N30_24	11	529	48,1	4250	2200	670	458
15X18_25	6	1080	180	4250	1860	458	460
15X22_26	7	3000	428,6	4250	1860	423	422
15Y30_27	6	244	40,7	4250	2120	670	458
16B18_28	14	1403	100,2	3770	1860	423	349
174N25_31	2	165	82,5	4250	1860	434	400
18SC18_36	5	708	141,6	2600	1860	394	383
214W18_42	8	3677	459,6	3760	1860	422	290
214Y18_43	5	3413	682,6	3760	1860	490	320
22A18_48	6	880	146,7	4250	1860	365	457
22G18_49	5	440	88,0	4250	1860	367	407
22X18_50	14	1307	93,4	4250	1860	365	307
23A25_51	5	396	79,2	4250	1860	585	308
23B25_52	29	2600	89,7	4250	1860	435	295
23C25_53	7	689	98,4	4250	1860	585	307
23N25_54	15	1085	72,3	4250	1860	435	295
23NATO18_55	4	476	119,0	3760	1860	774	440
23NATO25_56	4	740	185,0	3760	1860	810	465
23R25_57	6	550	91,7	4250	1860	585	400
26K19_58	2	365	182,5	2820	1870	577	572
26R16_59	2	315	157,5	2820	1870	730	595
28C18_60	7	226	32,3	4250	1860	708	434
28C25_61	5	294	58,8	4250	2200	498	414
28C30_62	8	188	23,5	4250	1860	590	456
28C38_63	2	58	29,0	4250	1860	694	608
28CC18_64	11	421	38,3	4250	1860	420	307
28R18_65	2	156	78,0	4250	1860	708	457
28R25_66	6	209	34,8	4250	1860	696	574
28R30_67	4	94	23,5	4250	1860	744	456
28T18_68	6	278	46,3	4250	2200	460	360
30A63_69	3	161	53,7	4250	1860	1704	297
30R1_70	1	985	985,0	4200	1860	540	900

Tabela 3.7 Instâncias da empresa *B (continuação da página anterior).*

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
35B22_71	10	390	39,0	4200	2130	861	391
37A18_73	1	2390	2390	4250	2070	760	490
37W18_74	4	100	25,0	4250	1860	391	325
38SC18_75	6	153	25,5	2820	1870	299	362
39H18_76	9	950	105,6	4250	1860	304	285
39N18_77	17	3781	222,4	3760	1860	370	340
39Q18_78	14	1144	81,7	4250	1860	273	340
39R18_79	6	465	77,5	4250	1860	423	340
40A63_80	2	231	115,5	4250	1860	548	416
40C1_81	4	242	60,5	4250	1860	548	416
40C2_82	2	242	121,0	4250	1860	450	415
40R1_83	4	349	87,3	4250	1860	548	416
42B18_84	8	394	49,3	4250	1860	442	260
42B22_85	9	237	26,3	4250	1860	540	255
42C18_86	22	2013	91,5	4250	1860	391	260
42C22_87	11	502	45,6	4250	1860	540	255
42F18_88	34	2500	73,5	4250	1860	300	325
42F22_89	18	710	39,4	4250	1860	480	255
42K18_90	3	209	69,7	4250	1860	439	331
42N18_91	17	1393	81,9	4250	1860	390	260
42N22_92	10	634	63,4	4250	1860	540	255
42O18_93	8	501	62,6	4250	1860	307	331
42R18_94	6	347	57,8	4250	1860	307	314
43L16_95	6	10185	1697,5	4050	2200	488	298
451W18_97	8	3605	450,6	3760	1860	422	290
451Y18_98	5	3413	682,6	3760	1860	490	320
45B16_99	16	5359	334,9	2820	2070	320	235
45C16_100	6	1734	289,0	2820	2070	326	270
45K16_101	6	1052	175,3	2820	1870	326	321
45R16_102	7	1575	225,0	2820	2070	326	306
58F18_103	18	3802	211,2	4250	1860	331	412
58G18_104	10	1240	124,0	4250	2120	331	250
58M18_105	13	1160	89,2	4250	1860	331	250
58O18_106	14	1456	104,0	4250	2120	331	250
64M18_107	7	2065	295,0	4250	1860	367	456
68M18_108	7	2065	295,0	4250	1860	367	456
68W18_109	9	3678	408,7	3760	1860	423	290
68Y18_110	6	3415	569,2	3760	1860	490	320
69W18_111	10	9090	909,0	3760	1860	422	290
69Y18_112	7	9033	1290,4	3760	1860	490	320
69Z18_113	8	6695	836,9	3760	2120	456	366
705MM18_114	7	3825	546,4	4250	1860	371	456
70M18_116	8	3886	485,8	4250	1860	367	456
70Z18_117	4	430	107,5	4250	1860	367	367
71R18_118	3	1324	441,3	4250	1860	457	371
71W18_119	6	2090	348,3	3760	1860	457	367
76R25_121	2	1260	630	3760	1860	522	388
86R25_122	9	1188	132,0	4250	1860	423	407
94H18_123	4	4935	1233,8	2700	1860	905	540
ANT39C_125	15	1200	80	4250	1860	367	306

Tabela 3.7 Instâncias da empresa *B (continuação da página anterior).*

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
ANT39R_126	17	1027	60,4	4250	1860	306	367
ANT41C_127	14	821	58,6	4250	1860	305	305
ARKEN18P_129	7	578	82,6	3760	1860	580	336
ARKEN25F_130	1	35	35,0	4250	2200	1211	432
ARKEN25P_131	1	400	400	3760	1860	1211	432
CARLET_132	14	1222	87,3	4250	1860	582	327
CUC28A_133	37	5654	152,8	4250	1860	364	260
GTM18AT_134	24	1428	59,5	4250	2120	325	357
KIT18N_135	15	320	21,3	4250	2200	298	301
MAC_136	1	32	32,0	4250	2120	808	808
MARAF1A_137	24	1213	50,5	4250	1860	419	256
MARAF1R_138	10	169	16,9	4200	2200	469	260
MOBILTUR_139	7	10710	1530	4250	2120	272	291
REVAL_140	60	5499	91,7	3760	1860	273	295
RIO18G_141	3	142	47,3	4250	2200	1251	449
RIO18N_142	25	4470	178,8	4250	2200	420	353
RIO25E_143	5	53	10,6	4250	1860	845	845
RIO25G_144	4	444	111,0	4200	2200	696	573
RIO25N_145	13	2056	158,2	4250	2200	626	240
STE1A_146	5	92	18,4	4250	1860	569	279
STE1R_147	2	33	16,5	4200	2200	730	538

Os outros dois conjuntos de instâncias foram adaptados de instâncias utilizadas para o problema de corte restrito onde apenas é considerada uma placa (Belov & Scheithaeur 2006, Cui 2008, Hifi 2001, Lodi & Monaci 2003). Neste problema, o objetivo é maximizar o lucro dos itens cortados de uma placa única. A cada item está associado um lucro e um limite superior no número de vezes que podem ser cortados da placa. As instâncias foram adaptadas para o problema de corte negligenciando o lucro associado a cada item e as procuras tomam o valor do limite superior associado a cada item. O primeiro conjunto (C) é composto por 30 instâncias com procuras médias entre 1 e 3 e o segundo conjunto (D) tem 20 instâncias com procuras médias entre 4 e 6.

Tabela 3.8: Instâncias da literatura *C*.

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
A1	19	62	3,3	50	60	9	11
A2	20	53	2,7	60	60	12	14
A3	20	46	2,3	70	80	15	14
A4	19	35	1,8	90	70	9	11
A5	20	45	2,3	132	100	13	12
CHL1	30	63	2,1	132	100	13	12
CHL2	10	19	1,9	62	55	11	9
CHL5	10	18	1,8	20	20	1	2
CHL6	30	65	2,2	130	130	18	12
CHL7	34	75	2,2	130	130	19	18
CU1	25	82	3,3	100	125	20	28
CU2	34	90	2,6	150	175	31	35
CW1	25	67	2,7	125	105	25	21
CW2	35	63	1,8	145	165	34	34
CW3	40	96	2,4	267	207	59	45
Hchl2	34	75	2,2	130	130	19	18
Hchl3s	10	51	5,1	127	98	15	13
Hchl4s	10	32	3,2	127	98	15	13
Hchl6s	22	60	2,7	253	244	35	38
Hchl7s	40	90	2,3	263	241	33	38
Hchl8s	10	18	1,8	49	20	1	2
Hchl9	35	76	2,2	65	76	10	10
HH	5	18	3,6	127	98	18	13
OF1	10	23	2,3	70	40	9	4
OF2	10	24	2,4	70	40	13	4
STS2	30	78	2,6	55	85	10	10
STS4	20	50	2,5	99	99	14	14
W	19	62	3,3	70	40	11	9
2	10	23	2,3	40	70	9	7
3	19	62	3,3	40	70	9	11

Tabela 3.9: Instâncias da literatura *D*.

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
ATP30	38	192	5,1	927	152	57	7
ATP31	51	258	5,1	856	964	44	50
ATP32	55	249	4,5	307	124	16	6
ATP33	44	224	5,1	241	983	15	52
ATP34	27	130	4,8	795	456	46	22
ATP35	29	153	5,3	960	649	50	34
ATP36	28	153	5,5	537	244	30	20
ATP37	43	222	5,2	440	881	23	51

Tabela 3.9 Instâncias da literatura *D (continuação da página anterior).*

Instância	Número de tipos de itens	Número de itens	Procura média	Largura da Placa	Altura da Placa	Menor largura do item	Menor altura do item
ATP38	40	202	5,1	731	358	41	19
ATP39	33	163	4,9	538	501	28	48
ATP40	56	289	5,2	683	138	34	6
ATP41	36	177	4,9	837	367	43	32
ATP42	58	325	5,6	167	291	8	21
ATP43	49	259	5,3	362	917	19	46
ATP44	39	196	5,0	223	496	11	29
ATP45	33	156	4,7	188	578	9	49
ATP46	42	197	4,7	416	514	23	40
ATP47	43	204	4,7	392	554	25	32
ATP48	34	167	4,9	931	254	47	18
ATP49	25	119	4,8	759	449	42	23

3.4.2 Comparação com modelos de afetação

Da tabela 3.10 à tabela 3.17 são apresentados o valor da solução obtida, isto é, o número de placas usadas (coluna z), o tempo de CPU em segundos para a construção do modelo e de otimização (coluna t), o tempo de CPU em segundos necessário apenas para construir o modelo (c), o limite inferior fornecido pela relaxação linear (coluna lb), o número de nodos da árvore de partição e avaliação sucessivas (coluna nbb) para os quatro problemas abordados: dois-estágios exato (2E), dois-estágios não-exato (2NE), três-estágios exato (3E) e três-estágios não-exato (3NE), para cada conjunto de instâncias, para o modelo proposto (modelo de corte) e os modelos de Lodi et al. (2004) (proposto para o problema 2NE) e de Puchinger & Raidl (2007) (proposto para o problema 3E). O limite de 7200 segundos foi estabelecido para o *solver Cplex* para todos os casos. Observou-se que, para todos os conjuntos de instâncias, o tempo de construção do modelo é sempre muito mais pequeno que o tempo gasto na sua otimização.

Sempre que a otimalidade de uma solução não possa ser comprovada, o va-

lor na coluna z é corresponde ao valor da solução incumbente, z, e o valor do gap absoluto $z - \lceil z_{LB} \rceil$, onde z_{LB} é um limite inferior para o valor ótimo é dado em parêntesis. O limite inferior é dado diretamente pelo método de partição e avaliação sucessivas do Cplex. No algoritmo de partição e avaliação sucessivas, cada nodo não explorado tem um limite inferior que corresponde ao valor ótimo da relaxação linear do seu pai. Um limite inferior global para o valor ótimo da solução inteira é o limite inferior mínimo de todos os nodos não explorados.

Instâncias reais

Para a empresa A (tabela 3.10 e tabela 3.11), numa análise preliminar, a solução ótima para instâncias com menos de 20 tipos de itens é obtida pelo modelo proposto em poucos segundos. Para instâncias com 20 - 40 tipos de itens a solução ótima é encontrada pelo modelo proposto em dezenas de segundos. As soluções ótimas para as instâncias com 40 a 100 tipos de itens são obtidas em centenas ou poucos milhares de segundos (para o problema 3E e 3NE algumas instâncias atingiram o tempo limite com soluções incumbentes). Para as instâncias com mais de 100 tipos de itens a otimalidade é provada apenas para as versões mais fáceis do problema (2E e 2NE), no entanto as soluções obtidas são ótimas ou muito próximas do ótimo. Apenas para a instância com mais tipos de itens e mais complexa (instância FA+AA-9_13, 3NE), não foi obtida nenhuma solução admissível pelo modelo proposto.

Neste conjunto de instâncias, o modelo proposto é claramente superior aos da literatura. Os modelos da literatura, em várias instâncias, atingem o limite de tempo sem provarem a otimalidade ou a otimização é interrompida porque não existe memória suficiente disponível. Os limites inferiores fornecidos pela relaxação linear do modelo proposto são muito melhores que os obtidos através

da relaxação linear dos modelos da literatura (melhoria de 12% para o problema 2NE e melhoria de 13% para o problema 3E). Uma qualidade superior dos limites inferiores contribui para um número menor de nodos no método de partição e avaliação sucessivas, necessário para obter uma solução ótima pelos modelos propostos.

Apenas em duas circunstâncias (FA+AA-9_13, 2NE e FA+AA-9_6, 3E) com as instâncias do subconjunto FA+AA, que tem procuras pequenas e um número grande de tipos de itens, os modelos da literatura obtiveram soluções melhores que os modelos propostos.

Tabela 3.10: Resultados para as instâncias da empresa A para o problema de dois-estágios.

Instância	$2E$					$2NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
AP-9_1	70	61,8	4,0	68,2	111	61	376,0	10,8	60,7	401	66 (8)	7209,0	8,1	54,7	624
AP-9_10	72	4,5	1,8	70,4	1	65	9,7	2,5	64,4	1	67 (3)	7203,3	3,1	59,3	5886
AP-9_11	65	12,1	3,5	64,2	1	58	18,8	5,0	57,2	1	59 (2)	7202,7	2,4	50,9	2808
AP-9_12	29	1,3	0,6	28,1	1	27	1,3	0,6	26,1	1	27(1)[a]	6674,5	0,2	22,4	42355
AP-9_13	31	110,4	2,4	30,1	201	28	25,5	2,8	27,2	1	30 (3)	7201,9	1,7	25,3	10742
AP-9_14	3	0	0	2,4	1	3	0	0	2,4	1	3	0	0	1,9	1
AP-9_15	16	0,1	0,1	14,3	1	14	0,2	0,1	12,9	1	14 (1)	7200,3	0,3	11,8	9796
AP-9_2	3	0,1	0,1	2,1	1	3	0,1	0,1	2,0	1	3	262,5	0	1,8	94641
AP-9_3	51	3,9	1,3	49,8	1	46	81,7	3,4	45,8	505	53 (8)	7207,5	6,7	43,3	500
AP-9_4	15	0	0	14,8	1	14	0	0	14,0	1	14	0,1	0	10,9	1
AP-9_5	15	0,7	0,4	14,1	1	14	0,9	0,5	13,5	1	16 (3)	7200,6	0,5	11,9	5665
AP-9_6	72	146,0	3,2	70,3	525	67	432,3	6,4	66,8	657	74 (8)	7210	9,3	63,1	5008
AP-9_7	42	0,5	0,4	41,0	1	39	0,9	0,4	38,9	41	41 (2)	7200,8	0,7	35,1	8893
AP-9_8	87	13,1	3,3	86,1	1	83	53,4	5,0	82,3	1	89 (7)	7207,5	7,3	77,9	7340
AP-9_9	5	0	0	4,7	1	5	0	0	4,7	1	5	6,4	0	4,5	1
AP-9-3MM-4MM_1	4	0	0	3,9	1	4	0	0	3,3	1	4	0,1	0	2,2	1
AP-9-3MM-4MM_2	37	0	0	36,1	1	36	0	0	36,0	1	36	0	0	22,4	1
AP-9-3MM-4MM_3	9	0	0	8,1	1	8	0	0	8,0	1	8	0	0	4,9	1
AP-9-3MM-4MM_4	3	0	0	2,7	1	3	0	0	2,7	1	3	0	0	2,2	1
AP-9-3MM-4MM_5	14	0,2	0,1	12,7	1	13	0,2	0,1	12,5	1	13	49,7	0,1	11,2	525
AP-9-3MM-4MM_6	2	0	0	1,9	1	2	0	0	1,9	1	2	0	0	1,2	1
AP-9-3MM-4MM_7	15	0	0	14,2	1	14	0	0	13,1	1	14	3,7	0	9,4	493
AP-9-3MM-4MM_8	2	0	0	1,1	1	2	0	0	1,1	1	2	0	0	1,0	1
FA+AA-9_1	43	4306,0	14,5	39,0	462	35	2275,1	45,3	34,3	131	36(1)[a]	6543,5	0,2	30,9	32240
FA+AA-9_10	10	41,3	4,0	8,0	1	8	46,9	4,6	7,5	1	8	3,0	0	6,8	401
FA+AA-9_11	34	1237,7	11,3	30,6	171	27	485,5	31,9	26,8	31	28(1)[a]	4521,9	0,1	24,3	39534
FA+AA-9_12	24	3229,8	8,8	21,0	572	19	639,2	14,8	18,3	31	19	356,0	0,1	16,9	181
FA+AA-9_13	44 (1)	7220,3	20	38,8	254	39 (4)	7363,5	163,3	34,6	66	37 (2)	7200,8	0,5	32,7	19334

Tabela 3.10 Resultados para as instâncias da empresa A para o problema de dois-estágios *(continuação da página anterior).*

Instância	$2E$					$2NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
FA+AA-9_14	7	4,5	1,2	5,8	1	6	3,6	1,3	5,2	1	6	0,3	0	4,6	1
FA+AA-9_15	21	633,6	6,5	18,4	201	17	330,4	18,3	16,4	61	18(1)[a]	3315,5	0,1	15,2	48477
FA+AA-9_2	23	1785,8	9,2	19,8	201	18	3380,7	26,2	17,0	371	19(1)[a]	4244,4	0,1	16,3	43484
FA+AA-9_3	52 (1)	7216,0	15,8	46,1	904	42 (1)	7278,6	78,1	40,8	301	43 (2)	7200,4	0,3	36,7	18959
FA+AA-9_4	10	35,8	3,8	7,6	1	8	36,2	4,5	7,0	1	8	2,3	0	6,4	59
FA+AA-9_5	25	90,1	7,2	21,4	1	20	292,0	13,6	19,4	201	21(1)[a]	4525,2	0	17,7	64183
FA+AA-9_6	25	1151,3	10	22,0	151	20	4333,0	42,5	19,2	201	21(1)[a]	5232,8	0,1	18,0	37061
FA+AA-9_7	15	204,0	6,9	12,4	31	12	829,9	15,2	10,8	181	12	1,5	0	10,3	1
FA+AA-9_8	33	2799,1	12,2	30,7	171	28	4197,0	39,3	27,3	201	29(1)[a]	5932,7	0,2	25,3	31254
FA+AA-9_9	6	8,4	2,4	3,8	1	4	74,6	3,0	3,6	111	4	0,2	0	3,4	1
FA+AP-9-10MM_1	11	0,6	0,5	9,9	1	9	0,7	0,5	8,9	1	9	1,8	0,6	6,5	1
FA+AP-9-10MM_2	6	0,1	0	5,1	1	5	0,1	0,1	4,6	1	5	5,9	0	4,2	1
FA+AP-9-10MM_3	26 (1)	7205,0	4,9	24,1	10743	23	41,8	5,5	22,1	1	23 (1)	7264,5	0,1	20,2	21559
FA+AP-9-10MM_4	4	0,9	0,4	3,3	1	4	0,7	0,4	3,0	1	4	0,1	0	2,8	16
TRAS-BC_1	16	18,0	2,7	14,7	31	15	17,7	3,6	14,1	11	15	6,9	0	13,0	259
TRAS-BC_2	20	1,7	0,6	17,5	1	17 (1)	7201,0	1,0	15,8	576040	17(1)[a]	5941,0	0,1	14,4	47680
TRAS-BC_3	22	0,4	0,3	19,9	1	19	2,1	1,5	18,5	1	19	0,3	0	14,2	1
TRAS-BC_4	8	0	0	7,2	1	8	0	0	7,2	1	8	0	0	4,5	1
TRAS-BC_5	7	0,1	0	6,5	1	7	0,1	0	6,4	1	7	0	0	4,8	1

Tabela 3.11: Resultados para as instâncias da empresa A para o problema de três-estágios.

Instância	$3E$					$3NE$									
	Modelo proposto					Modelo proposto					Puchinger et al. (2007)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
AP-9_1	59	166,0	22,7	58,3	1	77 (21)	7241,5	41,2	53,8	72	59	328,1	26,3	58,3	1
AP-9_10	65	10,2	3,5	64,3	1	70 (8)	7214,2	14,1	57,9	507	65	13,8	4,8	64,3	1
AP-9_11	56	22,8	6,3	55,7	1	63 (8)	7210,6	10,4	45,8	7047	56	27,4	7,9	55,2	1
AP-9_12	26	1,6	0,7	25,1	1	26 (1)	7200,9	0,8	22,0	20862	26	1,3	0,7	25,1	1
AP-9_13	27	50,4	3,2	26,6	91	35 (9)	7207,5	7,4	24,8	413	27	96,7	3,5	26,6	171
AP-9_14	3	0	0	2,4	1	3	0	0	1,9	1	3	0	0	2,4	1
AP-9_15	13	0,6	0,3	12,8	11	15 (2)	7201,5	1,3	11,5	8040	13	0,5	0,3	12,8	1
AP-9_2	2	0,2	0,1	2,0	4	2	5,4	0	1,8	642	2	0,2	0,1	2,0	4
AP-9_3	46	30,7	6,3	45,3	1	75 (31)	7234,8	34,4	42,0	195	46	38,0	6,5	45,3	1
AP-9_4	14	0	0	14,0	1	14	0,1	0	10,9	1	14	0	0	14,0	1
AP-9_5	14	0,9	0,6	13,4	1	17 (4)	7202,7	2,3	11,7	8125	14	1,0	0,6	13,3	1

[a]Cplex 11.0 retornou "falta de memória".

Tabela 3.11 Resultados para as instâncias da empresa A para o problema de três-estágios *(continuação da página anterior).*

Instância	$3E$										$3NE$				
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
AP-9_6	67	74,4	11,0	66,4	1	73 (7)	7246,2	46,0	62,1	494	67	126,7	11,3	66,4	1
AP-9_7	39	0,7	0,4	38,9	1	41 (2)	7203,3	3,2	34,7	8876	39	0,8	0,4	38,9	1
AP-9_8	83	29,6	6,2	82,2	1	88(6)[a]	6265,4	33,5	75,0	7894	83	23,2	6,4	82,1	1
AP-9_9	5	0	0	4,7	1	5	17,3	0,1	4,4	1	5	0	0	4,7	1
AP-9-3MM-4MM_1	4	0	0	3,3	1	4	0,3	0,1	2,2	1	4	0	0	3,3	1
AP-9-3MM-4MM_2	36	0	0	36,0	1	36	0	0	22,4	1	36	0	0	36,0	1
AP-9-3MM-4MM_3	8	0	0	8,0	1	8	0	0	4,9	1	8	0	0	8,0	1
AP-9-3MM-4MM_4	3	0	0	2,7	1	3	0	0	2,2	1	3	0	0	2,7	1
AP-9-3MM-4MM_5	13	0,2	0,2	12,3	1	13	85,1	0,3	11,2	524	13	0,2	0,2	12,3	1
AP-9-3MM-4MM_6	2	0	0	1,9	1	2	0	0	1,2	1	2	0	0	1,9	1
AP-9-3MM-4MM_7	14	0	0	13,1	1	14	12,2	0	9,4	494	14	0	0	13,1	1
AP-9-3MM-4MM_8	2	0	0	1,1	1	2	0	0	1,0	1	2	0	0	1,1	1
FA+AA-9_1	35	1497,2	66,7	34,0	31	36 (2)	7200,9	0,8	30,9	14105	35 (1)	7286,8	86,4	33,8	295
FA+AA-9_10	8	310,4	5,2	7,3	201	8	0,4	0	6,8	1	8	127,4	5,2	7,3	41
FA+AA-9_11	27	2346,1	54,6	26,1	71	27	258,9	0,5	24,3	1772	26	1365,0	66,5	25,9	1
FA+AA-9_12	19	2442,5	23,5	18,0	201	19 (1)	7203,0	0,4	16,9	45626	19 (1)	7234,9	34,6	17,8	628
FA+AA-9_13	36 (1)	7418,2	217,8	34,0	23	38 (4)	7202,6	2,4	32,6	9326	-	7637,4	436,7	33,3	1
FA+AA-9_14	6	4,7	1,7	5,1	1	6	0,4	0	4,6	1	6	3,2	1,6	5,1	1
FA+AA-9_15	17	193,5	25,6	16,1	1	17	1788,2	0,2	14,9	18171	17	1865,6	25,7	16,1	221
FA+AA-9_2	18 (1)	7249,2	48,9	16,8	341	19 (2)	7200,4	0,3	16,3	16986	19 (2)	7298,5	98,0	16,7	61
FA+AA-9_3	42 (1)	7333,8	133,2	40,5	241	42 (1)	7201,4	1,4	36,7	8053	42 (1)	7385,3	184,7	40,4	141
FA+AA-9_4	8	234,2	5,5	6,9	651	8(1)[a]	2505,3	0	6,4	125646	7	519,1	5,8	6,9	1236
FA+AA-9_5	20	1623,3	31,1	19,2	341	20	644,4	0,1	17,7	6902	20	3475,2	84,0	19,2	111
FA+AA-9_6	21 (2)	7338,0	137,3	18,8	105	20 (1)	7200,5	0,4	17,9	20206	20 (1)	7347,9	147,2	18,7	113
FA+AA-9_7	11	4483,6	30	10,6	201	12(1)[a]	2032,8	0,1	10,3	57525	12 (1)	7279,4	79,3	10,5	73
FA+AA-9_8	28	1605,3	75,5	26,9	81	29 (2)	7201,0	0,9	25,3	17160	29 (2)	7288,1	87,6	26,8	201
FA+AA-9_9	4	206,7	3,9	3,6	132	4	0,4	0	3,4	1	4	42,0	3,9	3,5	1
FA+AP-9-10MM_1	9	0,8	0,6	8,9	1	9	0,3	0	6,5	1	9	0,9	0,6	8,9	1
FA+AP-9-10MM_2	5	0,1	0,1	4,4	1	5	24,4	0	4,1	479	5	0,1	0,1	4,4	1
FA+AP-9-10MM_3	23	67,6	6,4	22,0	1	23 (1)	7200,5	0,3	20,2	29211	22	374,5	7,7	21,9	201
FA+AP-9-10MM_4	4	0,6	0,4	3,0	1	4	0,1	0	2,8	12	4	0,8	0,4	3,0	1
TRAS-BC_1	15	23,3	5,3	14,1	1	15	1,2	0	13,0	1	15	40,8	5,2	14,1	31
TRAS-BC_2	16	19,7	1,8	15,6	71	17 (1)	7200,4	0,3	14,3	21008	16	18,1	2,0	15,6	81
TRAS-BC_3	18	11,2	3,6	17,8	36	18	1,0	0	14,2	21	18	5,1	3,7	17,8	1
TRAS-BC_4	8	0	0	7,1	1	8	0	0	4,5	1	8	0	0	7,1	1
TRAS-BC_5	7	0,1	0,1	6,4	1	7	0	0	4,8	1	7	0,1	0,1	6,4	1

Para as instâncias B (tabela 3.12 e tabela 3.13), cerca de 90% das instâncias foram resolvidas pelo modelo proposto em menos de 10 segundos (93%, 90%,

[a]Cplex 11.0 retornou "falta de memória".

86% e 86% para 2E, 2NE, 3E e 3NE, respetivamente). Uma solução incumbente não foi encontrada para a instância $REVAL\_140$ para três tipos de problemas (2NE, 3E e 3NE). Do conjunto de instâncias B, esta instância é a detentora do maior número de tipos de itens. Com o modelo proposto, uma grande parte das instâncias deste conjunto foi resolvida na raiz da árvore de partição e avaliação sucessivas, confirmando deste modo a boa qualidade dos limites inferiores obtidos pela relaxação linear.

Tendo em consideração que nestas instâncias as procuras são elevadas, os modelos da literatura, embora sejam modelos compactos, são muito grandes e para várias instâncias não podem mesmo ser construídos. Numa análise grosseira, os modelos de Lodi et al. (2004) e de Puchinger & Raidl (2007) foram capazes de obter soluções ótimas em instâncias até 200 itens e soluções admissíveis em instâncias até 1500 itens. Para as instâncias com mais itens, os modelos tornam-se demasiado grandes para serem tratados computacionalmente.

Tabela 3.12: Resultados para as instâncias da empresa B para o problema de dois-estágios.

Instância	2E					2NE									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
03G18_1	76	0,3	0,1	75,0	1	76	0,3	0,2	75,0	1	78 (4)	7208,1	7,8	68,2	332
03J18_2	171	0,2	0,1	170,9	1	171	0,3	0,1	170,8	1	195 (29)	7221,0	19,8	149,8	224
03X22_3	110	0	0	108,9	1	110	0	0	108,9	1	1342(1250)[a]	174,1	43,0	87,9	1
10C30_4	24	0	0	22,9	1	21	0	0	20,7	1	21	227,5	0	17,8	8187
10M30_5	12	0	0	11,4	1	11	0	0	10,5	1	11	11,0	0	8,9	261
10W30_6	10	0	0	9,7	1	10	0	0	9,7	1	10	0,1	0	8,9	1
114M18_7	548	0	0	547,5	1	537	0	0	536,3	1	[b]	[b]	[b]	[b]	[b]
114UU18_9	347	0	0	346,4	1	334	0	0	333,9	1	[a]	300,2	290	a	1
12D16_10	86	8,7	0,5	84,9	517	85	265,3	10,3	84,2	1	[a]	327,7	318,2	a	1
12F18_11	79	16,5	2,3	77,9	1	78	61,1	3,8	77,7	1	[b]	[b]	[b]	[b]	[b]
12M18_12	274	6,3	1,5	273,5	1	271	60,1	8,1	270	1	[b]	[b]	[b]	[b]	[b]
146O18_17	466	0	0	465,5	1	465	0,1	0	465,0	1	[b]	[b]	[b]	[b]	[b]
146OO18_18	231	0,1	0,1	230,3	1	230	0,3	0,2	229,1	1	[b]	[b]	[b]	[b]	[b]
15A30_19	37	0	0	36,1	1	35	0	0	34,3	1	36 (5)	7200,9	0,6	29,6	8979
15B30_20	18	0	0	16,8	1	16	0	0	15,4	1	16	6,3	0	12,1	261
15G30_21	25	0	0	23,3	1	24	0,1	0	23,3	1	27 (4)	7200,7	0,6	21,7	7890
15J30_22	12	0	0	11,1	1	12	0	0	11,1	1	12	2,1	0	7,6	486

Tabela 3.12 Resultados para as instâncias da empresa B para o problema de dois-estágios *(continuação da página anterior).*

Instância	2E					2NE									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
15M30_23	14	0	0	13,1	1	14	0	0	13,1	1	14	10,8	0,1	11,9	1
15N30_24	50	0	0	48,7	1	47	0,1	0,1	46,2	1	50 (4)	7203,0	2,7	43,4	1279
15X18_25	107	0	0	106,9	1	95	0	0	95,0	1	101 (6)	7222,7	22,3	84,4	51
15X22_26	229	0	0	228,3	1	229	0,1	0,1	228,3	1	[b]	[b]	[b]	[b]	[b]
15Y30_27	18	0	0	16,6	1	17	0	0	16,5	1	19 (3)	7200,3	0,3	15,1	11719
16B18_28	86	0,1	0,1	85,1	1	85	0,4	0,2	84,3	1	1403 (1322)b	404,7	48,9	80,8	1
174N25_31	10	0	0	8,8	1	9	0	0	8,8	1	9	311,0	0,1	8,3	5208
18SC18_36	48	0	0	47,2	1	47	0	0	46,8	1	48 (2)	7206,9	6,4	40,1	468
214W18_42	129	0,1	0,1	128,3	1	128	0,2	0,1	127,7	1	[b]	[b]	[b]	[b]	[b]
214Y18_43	221	0	0	220,8	1	221	0	0	220,5	1	[b]	[b]	[b]	[b]	[b]
22A18_48	38	0,1	0,1	37,5	1	38	0,2	0,2	37,0	1	62 (26)	7212,5	12,1	36,0	379
22G18_49	19	0	0	18,2	1	19	0,1	0	18,2	1	21 (3)	7202,0	1,6	17,0	877
22X18_50	58	2,3	0,8	56,8	1	57	20,5	2,2	56,8	1	125 (70)	186,4	39,4	54,9	1
23A25_51	34	0	0	33,3	1	30	0	0	29,3	1	31 (3)	7201,6	1,1	25,6	3756
23B25_52	221(1)[a]	1213,0	0,8	219,3	92955	192	3,2	1,2	191,1	1	[a]	318,8	308,1	[b]	1
23C25_53	57	0,1	0	55,7	1	52	0,1	0	50,9	1	57 (9)	7206,4	6,1	44,8	185
23N25_54	97	0,4	0,3	96,5	1	83	0,5	0,4	82,7	1	111(32)	7223,0	22,5	67,8	1
23NATO18_55	96	0	0	95,8	1	96	0	0	95,8	1	96 (1)	7202,0	2,0	85,3	3145
23NATO25_56	97	0	0	95,8	1	96	0	0	95,8	1	97 (2)	7207,5	7,2	87,4	650
23R25_57	61	0	0	60,3	1	53	0	0	52,2	1	54 (5)	7203,3	3,0	44,2	2668
26K19_58	35	0	0	34,6	1	35	0	0	34,6	1	35	526,4	0,9	26,9	521
26R16_59	35	0	0	35,0	1	35	0	0	35,0	1	35	38,3	0,6	27,0	1
28C18_60	19	0,2	0,2	18,9	1	19	0	0	18,6	1	20 (2)	7200,2	0,2	17,3	9466
28C25_61	18	0,1	0	16,9	1	17	0	0	16,7	1	19 (2)	7200,5	0,5	14,5	20712
28C30_62	16	0	0	15,4	1	16	0	0	15,4	1	16(1)[a]	2803,7	0,1	14,5	34935
28C38_63	6	0	0	5,3	1	6	0	0	5,3	1	6	0,5	0	4,5	1
28CC18_64	24	0,3	0,2	23,9	1	24	0,6	0,4	23,3	1	25 (2)	7201,9	1,4	22,0	5773
28R18_65	11	0	0	10,2	1	11	0	0	10,2	1	11	2,7	0,1	9,7	1
28R25_66	28	0	0	27,4	1	28	0	0	27,0	1	28(1)[a]	5721,8	0,2	22,1	32358
28R30_67	8	0	0	7,9	1	8	0	0	7,9	1	8	1,7	0	7,4	1
28T18_68	14	0	0	13,7	1	14	0,1	0	13,4	1	15 (1)	7200,5	0,4	13,2	8758
30A63_69	21	0	0	20,1	1	21	0	0	20,1	1	21	3,0	0,1	16,0	1
30R1_70	71	0	0	70,4	1	71	0	0	70,4	1	71	1081,0	16,8	61,3	1
35B22_71	67	0	0	65,7	1	63	0	0	62,1	1	67 (7)	7201,3	1,1	52,2	7899
37A18_73	120	0	0	119,5	1	120	0	0	119,5	1	[a]	294,8	239,2	[b]	1
37W18_74	3	0,1	0	2,8	1	3	0	0	2,8	11	3	8,8	0	2,4	181
38SC18_75	14	0	0	12,4	1	13	0,1	0	12,2	1	13	5459,8	0,1	10,9	49560
39H18_76	54	0,1	0,1	53,6	1	51	0,5	0,3	49,9	1	61 (13)	7215,5	15,1	47,1	19
39N18_77	218	1,8	0,7	217,5	1	214	8,5	1,1	213,8	81	[b]	[b]	[b]	[b]	[b]
39Q18_78	47	2,2	0,8	46,2	1	44	23,9	1,0	43,9	496	90(48)[a]	1191,2	26,4	41,8	1
39R18_79	21	0,1	0,1	20	1	20	0,1	0,1	19,1	1	21 (2)	7202,0	1,8	18,0	6111
40A63_80	12	0	0	11,1	1	11	0	0	10,4	1	11	613,3	0,2	9,1	717
40C1_81	14	0	0	13,0	1	13	0	0	12,2	1	13 (1)	7200,3	0,3	11,3	19205
40C2_82	6	0	0	5,3	1	6	0	0	5,3	1	6(1)[a]	3376,3	0,1	4,7	44648
40R1_83	20	0	0	18,8	1	18	0	0	17,5	1	21 (3)	7200,9	0,8	16,3	8974
42B18_84	51	0,1	0	50,5	1	38	0,1	0,1	37,8	11	39 (3)	7201,2	1,1	34,0	4591
42B22_85	20	0,5	0,2	19,4	1	20	0,4	0,3	19,3	1	21 (1)	7204,4	0,5	18,4	21053

Tabela 3.12 Resultados para as instâncias da empresa B para o problema de dois-estágios *(continuação da página anterior).*

Instância	2E					2NE									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
42C18_86	206	0,5	0,3	205,3	1	172	2,8	1,3	171,6	1	[a]	329,2	144,8	160,3	1
42C22_87	43	0,1	0,1	42,0	1	41	0,2	0,1	40	1	43 (3)	7202,5	2,3	36,0	5301
42F18_88	326	1,3	0,8	324,5	1	262	54,0	11,8	261,0	1	[a]	284,2	274,5	[b]	1
42F22_89	51	1,0	0,5	50,3	1	49	3,3	0,7	48,9	21	55 (6)	7206,7	6,4	46,1	1257
42K18_90	21	0	0	20,4	1	20	0	0	19,9	1	21 (1)	7200,3	0,2	18,3	21217
42N18_91	141	0,9	0,5	139,9	1	117	1,9	0,9	116,2	1	189(79)[a]	264,5	47,8	108,3	1
42N22_92	74	0,9	0,1	73,7	1	66	0,4	0,2	65,1	1	70 (6)	7214,2	13,8	60,7	41
42O18_93	35	0	0	34,4	1	34	0,1	0,1	33,8	1	38 (5)	7202,6	2,3	32,3	956
42R18_94	20	2,7	0,8	18,8	1	19	0,1	0,1	18,3	1	21 (3)	7201,0	0,8	17,0	4226
43L16_95	310	0	0	309,3	1	309	0,1	0,1	308,6	1	[b]	[b]	[b]	[b]	[b]
451W18_97	127	0,5	0,5	126,5	1	126	0,2	0,1	125,9	1	[b]	[b]	[b]	[b]	[b]
451Y18_98	221	0	0	220,8	1	221	0	0	220,5	1	[b]	[b]	[b]	[b]	[b]
45B16_99	215	1,8	0,8	213,9	1	208	2,8	1,1	207,0	1	[b]	[b]	[b]	[b]	[b]
45C16_100	78	0,1	0	77,3	1	73	0,1	0	72,5	1	1734[a]	271,6	93,4	66,2	1
45K16_101	51	0	0	50,2	1	50	0,1	0,1	49,5	1	62 (14)	7221,2	20,6	44,9	5
45R16_102	58	0,1	0,1	57,4	1	57	0,4	0,1	56,9	71	1575[a]	130,2	71,3	54,3	1
58F18_103	466	0,2	0,1	465,0	1	425	0,5	0,4	424,0	1	[b]	[b]	[b]	[b]	[b]
58G18_104	121	0,2	0,2	120,8	1	105	0,3	0,2	104,8	1	1240(1143)[a]	222,5	35,6	91,4	1
58M18_105	103	0,2	0,2	101,8	1	97	0,4	0,3	96,5	1	159(68)[a]	729,6	27,7	89,4	1
58O18_106	105	0,5	0,3	103,8	1	103	0,4	0,3	102,2	1	1456(1361)[a]	386,0	54,8	94,5	1
64M18_107	106	0,5	0,2	105,3	1	106	0,8	0,4	105,0	1	[a]	226,3	159,5	101,6	1
68M18_108	106	0,3	0,2	105,3	1	106	0,8	0,4	105,0	1	[a]	225,4	159,3	101,6	1
68W18_109	129	0,1	0,1	128,4	1	128	0,2	0,1	127,7	1	[b]	[b]	[b]	[b]	[b]
68Y18_110	227	0	0	226,8	1	226	0	0	225,6	1	[b]	[b]	[b]	[b]	[b]
69W18_111	355	0,2	0,1	354,8	1	353	0,2	0,2	352,4	1	[b]	[b]	[b]	[b]	[b]
69Y18_112	856	0	0	855,4	1	841	0,1	0	840,4	1	[b]	[b]	[b]	[b]	[b]
69Z18_113	318	0,1	0,1	317,5	1	317	0,2	0,1	316,5	1	[b]	[b]	[b]	[b]	[b]
705MM18_114	176	0,3	0,2	175,2	1	176	0,6	0,3	175,2	1	[b]	[b]	[b]	[b]	[b]
70M18_116	201	0,1	0,1	200,1	1	201	0,8	0,4	200,1	1	[b]	[b]	[b]	[b]	[b]
70Z18_117	12	0	0	11,3	1	12	0	0	11,1	1	12 (1)	7201,6	1,5	10,3	2698
71R18_118	60	0	0	59,6	1	60	0	0	59,6	1	1324(1271)[a]	279,6	40,9	52,3	1
71W18_119	124	0	0	123,8	1	124	0	0	123,8	1	[a]	222,5	162,0	109,7	1
76R25_121	65	0	0	65,0	1	60	0	0	60	1	1260(1205)[a]	185,9	35,1	54,7	1
86R25_122	68	0,1	0,1	67,8	1	68	0,4	0,2	67,0	1	1188(1130)[a]	169,4	30,5	57,3	1
94H18_123	1130	0,2	0,2	1129,2	1	1095	0	0	1095,0	1	[b]	[b]	[b]	[b]	[b]
ANT39C_125	60	3,9	0,3	58,9	509	59	6,2	1,1	58,6	1	1218(1161)[a]	152,6	31,8	56,8	1
ANT39R_126	61(1)[b]	6455,5	0,5	58,3	1621088	59	28,5	1,7	58,3	201	73(18)[a]	6277,3	19,1	51,9	512
ANT41C_127	47	0,3	0,2	46,5	1	46	0,9	0,5	45,2	1	56 (12)	7210,2	9,8	43,1	142
ARKEN18P_129	42	0	0	40,4	1	41	0,1	0,1	39,9	1	43 (4)	7204,8	3,6	36,5	10123
ARKEN25F_130	3	0	0	2,3	1	3	0	0	2,3	1	3	0,1	0	2,0	1
ARKEN25P_131	34	0	0	33,3	1	34	0	0	33,3	1	34	136,7	1,2	29,9	11
CARLET_132	80	0,5	0,2	78,7	1	71	0,6	0,4	70,2	1	1222(1157)[a]	185,1	32,4	64,7	1
CUC28A_133	302(1)[a]	6383,4	1,4	300,4	366966	300	199,0	16,4	299,0	1	a	a	a	a	a
GTM18AT_134	188(1)[a]	2950,2	0,3	186,4	2	185(1)[a]	1807,7	1,0	183,7	182435	1428(1249)[a]	317,0	51,6	169,3	1
KIT18N_135	31	3,9	1,8	29,9	1	26	1,9	1,2	25,1	1	27 (2)	7200,7	0,6	21,1	8980
MAC_136	4	0	0	3,2	1	4	0	0	3,2	1	4	0,1	0	2,3	1
MARAF1A_137	59(1)[a]	2721,3	1,0	57,6	172354	56	52,6	9,4	55,2	1	1213(1161)[b]	163,9	31,5	51,1	1

Tabela 3.12 Resultados para as instâncias da empresa B para o problema de dois-estágios *(continuação da página anterior).*

Instância	$2E$					$2NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
MARAF1R_138	9	0,7	0,6	8,2	1	9	0,7	0,5	8,1	1	9 (1)	7200,1	0,1	7,0	34512
MOBILTUR_139	231	0,2	0,2	230,4	1	231	0,1	0,1	230,4	1	a	a	a	a	a
REVAL_140	286	1562,5	4,8	284,6	7001	[b]	7263,4	62,8	280	91	a	a	*b*	*b*	*b*
RIO18G_141	15	0	0	14,2	1	15	0	0	14,2	1	15	233,4	0	10	4506
RIO18N_142	196(1)[a]	1573,3	1,3	194,0	36366	192	34,3	5,3	191,3	1	*b*	*b*	*b*	*b*	*b*
RIO25E_143	6	0	0	5,3	1	6	0	0	5,3	1	6	0,2	0	4,8	1
RIO25G_144	38	0	0	37,6	1	36	0	0	35,3	1	36 (1)	7201,8	1,6	32,3	7297
RIO25N_145	146	0,7	0,6	144,9	1	144	0,6	0,2	143,3	1	[a]	391,6	153,5	138,4	1
STE1A_146	6	0,3	0,3	5,0	1	5	0	0	4,7	1	5	32,0	0	4,2	690
STE1R_147	3	0	0	2,1	1	3	0	0	2,1	1	3	877,5	0	1,5	411774

Tabela 3.13: Resultados para as instâncias da empresa B para o problema de três-estágios.

Instância	$3E$					$3NE$									
	Modelo proposto					Modelo proposto					Puchinger et al. (2007)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
03G18_1	76	1,8	0,2	75,0	488	80 (6)	7235,5	35,1	64,9	389	75	0,7	0,3	74,8	51
03J18_2	169	0,3	0,2	168,5	1	[a]	7292,6	91,0	142,8	1	169	0,3	0,2	168,5	1
03X22_3	108	0	0	107,9	1	1342[a]	335,1	196,1	85,4	1	108	0	0	107,9	1
10C30_4	19	0	0	18,7	1	19	3695,4	0,1	17,8	45321	19	0	0	18,7	1
10M30_5	11	0	0	10,5	1	11	9,4	0,1	8,7	141	11	0	0	10,5	1
10W30_6	9	0	0	9,0	1	9	0,1	0	8,9	1	9	0	0	9,0	1
114M18_7	537	0	0	536,3	1	*b*	*b*	*b*	*b*	*b*	537	0	0	536,3	1
114UU18_9	334	0	0	333,9	1	*b*	*b*	*b*	*b*	*b*	334	0	0	333,9	1
12D16_10	85	234,2	14,4	84,0	1	*b*	*b*	*b*	*b*	*b*	85	132,8	14,3	84,0	1
12F18_11	78	350,5	14,3	77,7	1	*b*	*b*	*b*	*b*	*b*	78	348,2	14,2	77,7	1
12M18_12	270	873,1	15,8	269,9	465	*b*	*b*	*b*	*b*	*b*	270	1221,5	16,3	269,9	401
146O18_17	465	0,1	0,1	465,0	1	*b*	*b*	*b*	*b*	*b*	465	0,1	0,1	465,0	1
146OO18_18	230	0,7	0,4	229,1	1	*b*	*b*	*b*	*b*	*b*	230	0,7	0,4	229,1	1
15A30_19	35	0	0	34,3	1	36 (5)	7202,8	2,7	28,9	9513	35	0	0	34,3	1
15B30_20	16	0	0	15,4	1	16	6,4	0,1	12,1	261	16	0	0	15,4	1
15G30_21	24	0,1	0,1	23,3	1	27 (4)	7202,8	2,7	21,1	7357	24	0,1	0,1	23,1	1
15J30_22	12	0	0	11,1	1	12	2,1	0	7,4	482	12	0	0	11,1	1
15M30_23	14	0	0	13,1	1	14(0)[a]	11,0	0,2	11,7	1	14	0	0	13,1	1

[a]Cplex 11.0 retornou ”falta de memória”.

[b]Memória insuficiente para construir o modelo.

Tabela 3.13 Resultados para as instâncias da empresa B para o problema de três-estágios *(continuação da página anterior).*

Instância	3E										3NE				
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
15N30_24	47	0,2	0,1	46,2	1	50 (4)	7212,3	12,1	41,5	916	47	0,2	0,1	46,2	1
15X18_25	95	0	0	95,0	1	138(43)[a]	2616,0	102,2	78,6	1	95	0	0	95,0	1
15X22_26	229	0,3	0,2	228,3	1	[b]	[b]	[b]		[b]	229	0,3	0,2	228,3	1
15Y30_27	17	0,1	0	16,5	1	19 (3)	7201,3	1,2	14,8	9896	17	0,1	0	16,5	1
16B18_28	85	0,7	0,4	84,3	1	1403b	352,9	224,1	78,6	1	85	0,7	0,5	84,3	1
174N25_31	9	0	0	8,8	1	9	3145,4	0,3	8,2	46249	9	0	0	8,8	1
18SC18_36	47	0	0	46,7	1	[a]	7229,6	28,8	36,5	1	47	0	0	46,7	1
214W18_42	128	0,4	0,3	127,7	1	[b]	[b]	[b]	[b]	[b]	128	0,4	0,3	127,7	1
214Y18_43	221	0	0	220,5	1	[b]	[b]	[b]	[b]	[b]	221	0	0	220,5	1
22A18_48	38	0,5	0,3	37,0	1	62 (26)	7255,7	55,1	33,4	380	38	0,5	0,3	37,0	1
22G18_49	19	0,1	0,1	18,2	1	21 (3)	7207,5	7,1	16,0	886	19	0,1	0,1	18,2	1
22X18_50	57	123,1	1,8	56,8	1385	985(30)[a]	372,1	181,0	51,3	1	57	125,6	1,8	56,8	1385
23A25_51	30	0	0	29,3	1	31 (3)	7205,3	5,1	23,9	3772	30	0	0	29,3	1
23B25_52	192	10,8	1,4	191,1	201	[b]	[b]	[b]	[b]	[b]	192	11,0	1,4	191,1	201
23C25_53	52	0,1	0	50,9	1	56 (8)	7227,8	27,3	40	161	52	0,1	0	50,9	1
23N25_54	83	1,1	0,5	82,7	1	113(38)[a]	1517,4	103,4	66,2	1	83	1,1	0,5	82,7	1
23NATO18_55	96	0	0	95,8	1	96 (1)	7208,9	8,8	83,1	4772	96	0	0	95,8	1
23NATO25_56	96	0	0	95,8	1	97 (2)	7233,3	32,8	77,0	538	96	0	0	95,8	1
23R25_57	53	0	0	52,2	1	54 (5)	7213,9	13,5	41,7	3310	53	0	0	52,2	1
26K19_58	35	0	0	34,6	1	35	633,6	4,0	24,9	552	35	0	0	34,6	1
26R16_59	35	0	0	35,0	1	35	40,8	2,6	23,2	1	35	0	0	35,0	1
28C18_60	19	0	0	18,6	1	20 (2)	7201,1	1,0	17,0	9717	19	0	0	18,6	1
28C25_61	17	0	0	16,7	1	19 (2)	7202,3	2,1	13,9	20720	17	0	0	16,7	1
28C30_62	16	0,1	0,1	15,4	1	16(1)[a]	2795,2	0,5	14,5	34911	16	0,1	0,1	15,4	1
28C38_63	6	0	0	5,3	1	6	0,5	0	4,3	1	6	0	0	5,3	1
28CC18_64	24	1,5	0,8	23,2	1	25 (2)	7206,4	6,1	21,4	2486	24	1,5	0,8	23,2	1
28R18_65	11	0	0	10,2	1	11	3,0	0,3	9,4	1	11	0	0	10,2	1
28R25_66	28	0	0	27,0	1	28(1)[a]	5649,8	0,8	21,7	32107	28	0	0	27,0	1
28R30_67	8	0	0	7,9	1	8	1,8	0	7,3	1	8	0	0	7,9	1
28T18_68	14	0,1	0,1	13,4	1	15 (1)	7201,8	1,8	12,8	8817	14	0,1	0,1	13,4	1
30A63_69	21	0	0	20,1	1	21	3,1	0,3	15,7	1	21	0	0	20,1	1
30R1_70	71	0	0	70,4	1	71	1154,5	77,5	41,7	1	71	0	0	70,4	1
35B22_71	63	0,1	0	62,1	1	65 (5)	7205,1	4,9	48,7	9661	63	0,1	0	62,1	1
37A18_73	120	0	0	119,5	1	[b]	[b]	[b]	[b]	[b]	120	0	0	119,5	1
37W18_74	3	0	0	2,8	8	3	14,3	0,1	2,3	307	3	0	0	2,8	8
38SC18_75	13	0,1	0,1	12,2	1	13(1)[a]	6452,2	0,3	10,9	37723	13	0,1	0,1	12,2	1
39H18_76	50	2,5	1,2	49,8	1	61 (13)	7270	69,3	43,1	12	50	2,4	1,2	49,8	1
39N18_77	214	10,9	3,3	213,8	1	[b]	[b]	[b]	[b]	[b]	214	11,0	3,3	213,8	1
39Q18_78	44	82,6	3,3	43,9	468	924(882)[a]	274,0	121,3	39,1	1	44	83,8	3,3	43,9	468
39R18_79	20	0,2	0,1	19,1	1	21 (2)	7208,5	8,3	16,7	7140	20	0,2	0,2	19,1	1
40A63_80	11	0	0	10,4	1	11	595,6	1,0	8,6	749	11	0	0	10,4	1
40C1_81	13	0	0	12,2	1	13 (1)	7201,3	1,3	10,9	18992	13	0	0	12,2	1
40C2_82	6	0	0	5,3	1	6(1)[a]	3193,2	0,3	4,5	44567	6	0	0	5,3	1
40R1_83	18	0	0	17,5	1	21 (3)	7203,6	3,5	15,6	8999	18	0	0	17,5	1
42B18_84	38	0,2	0,1	37,8	1	40 (4)	7205,1	5,0	32,9	3482	38	0,2	0,1	37,8	1
42B22_85	20	0,5	0,3	19,3	1	21 (1)	7202,2	2,1	18,0	21841	20	0,5	0,3	19,3	1
42C18_86	172	6,5	2,4	171,4	1	[a]	700,8	686,0	[b]	1	172	7,6	2,4	171,4	1

Tabela 3.13 Resultados para as instâncias da empresa B para o problema de três-estágios *(continuação da página anterior).*

Instância	$3E$										$3NE$				
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
42C22_87	40	0,3	0,2	39,5	1	45 (7)	7210,6	10,4	34,9	559	40	0,2	0,2	39,2	1
42F18_88	260	386,8	24,7	259,3	1	[b]	[b]	[b]	[b]	[b]	260	147,8	25,0	259,3	1
42F22_89	49	2,4	1,2	48,1	1	57 (9)	7229,3	29,0	44,6	290	49	3,5	1,2	48,1	1
42K18_90	20	0	0	19,9	1	21 (1)	7201,0	0,8	17,7	21208	20	0	0	19,9	1
42N18_91	116	20,1	2,0	115,9	359	1302(1192)[a]	533,1	219,3	104,4	1	116	20,4	2,0	115,9	359
42N22_92	65	0,5	0,3	64,6	1	71 (8)	7261,3	60,7	58,0	44	65	0,5	0,3	64,5	1
42O18_93	34	0,3	0,2	33,8	1	38 (5)	7210,4	10,2	31,1	927	34	0,3	0,2	33,8	1
42R18_94	19	0,2	0,1	18,3	1	21 (3)	7203,7	3,5	16,5	4232	19	0,2	0,1	18,3	1
43L16_95	309	0,1	0,1	308,6	1	[b]	[b]	[b]	[b]	[b]	309	0,1	0,1	308,6	1
451W18_97	126	0,5	0,3	125,9	1	[b]	[b]	[b]	[b]	[b]	126	0,5	0,3	125,9	1
451Y18_98	221	0	0	220,5	1	[b]	[b]	[b]	[b]	[b]	221	0	0	220,5	1
45B16_99	208	7,7	1,7	207,0	1	[b]	[b]	[b]	[b]	[b]	208	7,8	1,8	207,0	1
45C16_100	73	0,1	0,1	72,5	1	[a]	673,7	428,6	60,2	1	73	0,1	0,1	72,5	1
45K16_101	50	0,2	0,1	49,5	1	88(40)[a]	2238,0	94,3	41,1	1	50	0,2	0,1	49,5	1
45R16_102	57	1,4	0,2	56,9	341	1323b	413,8	328,0	49,2	1	57	1,4	0,2	56,9	341
58F18_103	424	1,1	0,7	423,1	1	[b]	[b]	[b]	[b]	[b]	424	1,2	0,8	423,1	1
58G18_104	105	0,4	0,3	104,1	1	1080(983)[a]	496,8	154,8	86,9	1	104	0,5	0,3	103,6	1
58M18_105	97	1,0	0,5	96,3	1	1087(997)[a]	423,1	126,5	86,2	1	97	1,0	0,5	96,3	1
58O18_106	103	0,6	0,4	102,2	1	1383[a]	393,2	250,7	88,3	1	103	0,6	0,5	102,2	1
64M18_107	106	1,0	0,5	105,0	1	[a]	777,9	754,1	[b]	1	106	1,0	0,5	105,0	1
68M18_108	106	1,0	0,5	105,0	1	[b]	751,7	742,0	[b]	1	106	1,0	0,5	105,0	1
68W18_109	128	0,5	0,3	127,7	1	[b]	[b]	[b]	[b]	[b]	128	0,5	0,3	127,7	1
68Y18_110	226	0	0	225,6	1	[b]	[b]	[b]	[b]	[b]	226	0	0	225,6	1
69W18_111	353	0,6	0,4	352,4	1	[b]	[b]	[b]	[b]	[b]	353	0,6	0,4	352,4	1
69Y18_112	841	0,1	0,1	840,4	1	[b]	[b]	[b]	[b]	[b]	841	0,1	0,1	840,4	1
69Z18_113	317	0,3	0,2	316,5	1	[b]	[b]	[b]	[b]	[b]	317	0,3	0,2	316,5	1
705MM18_114	176	0,6	0,3	175,2	1	[b]	[b]	[b]	[b]	[b]	176	0,7	0,3	175,2	1
70M18_116	201	3,1	1,0	200,1	1	[b]	[b]	[b]	[b]	[b]	201	3,1	1,0	200,1	1
70Z18_117	12	0,1	0	11,1	1	12 (1)	7206,8	6,6	9,7	2708	12	0,1	0,1	11,1	1
71R18_118	60	0	0	59,6	1	1324(1271)[a]	442,3	188,9	44,0	1	60	0	0	59,6	1
71W18_119	124	0,1	0	123,8	1	[a]	762,5	752,6	[b]	1	124	0	0	123,8	1
76R25_121	60	0	0	60	1	1260(1151)[a]	323,3	162,3	48,4	1	60	0	0	60	1
86R25_122	68	0,3	0,2	67,0	1	949(891)[a]	331,3	136,0	50,2	1	68	0,3	0,2	67,0	1
94H18_123	1095	0	0	1095,0	1	[b]	[b]	[b]	[b]	[b]	1095	0	0	1095,0	1
ANT39C_125	59	2,4	1,2	58,6	1	888(831)[a]	315,8	146,5	52,0	1	59	2,4	1,2	58,6	1
ANT39R_126	59	68,3	5,8	58,3	1	93 (38)	7288,0	87,6	49,0	512	59	47,5	5,8	58,3	1
ANT41C_127	46	12,6	2,6	45,2	21	53 (9)	7245,2	44,8	39,6	70	46	11,4	2,6	45,2	1
ARKEN18P_129	40	0,2	0,1	39,8	1	45 (8)	7216,3	16,0	35,4	448	40	0,2	0,1	39,8	1
ARKEN25F_130	3	0	0	2,3	1	3	0,1	0	2,0	1	3	0	0	2,3	1
ARKEN25P_131	34	0	0	33,3	1	34	114,0	5,3	24,6	1	34	0	0	33,3	1
CARLET_132	70	0,9	0,5	69,1	1	1219(1154)[a]	410,9	147,8	62,2	1	70	0,9	0,5	69,1	1
CUC28A_133	300	484,2	32,5	299,0	1	[b]	[b]	[b]	[b]	[b]	300	596,3	32,7	299,0	1
GTM18AT_134	181(1)[a]	2816,7	3,8	179,8	91284	1427 (1253)b	706,8	250,8	165,8	1	180	3561,5	3,9	179,8	38074
KIT18N_135	25	11,9	3,9	24,1	1	26 (2)	7202,8	2,7	21,0	9006	24	13,0	3,8	23,6	1
MAC_136	4	0	0	3,2	1	4	0,1	0	2,3	1	4	0	0	3,2	1
MARAF1A_137	56	487,8	28,1	55,1	1	1210(1158)[a]	413,2	151,1	50	1	56	351,0	28,2	55,1	1
MARAF1R_138	9	1,2	0,8	8,1	1	9 (1)	7200,5	0,4	6,8	19218	9	1,2	0,8	8,1	1

Tabela 3.13 Resultados para as instâncias da empresa B para o problema de três-estágios *(continuação da página anterior).*

Instância	$3E$										$3NE$				
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
MOBILTUR_139	231	0,2	0,1	230,4	1	[b]	[b]	[b]	[b]	[b]	231	0,2	0,1	230,4	1
REVAL_140	[b]	7397,3	197,0	276,9	1	[b]	[b]	[b]	[b]	[b]	[a]	7398,0	197,5	276,8	1
RIO18G_141	15	0	0	14,2	1	15	245,9	0,2	10	4542	15	0	0	14,2	1
RIO18N_142	192	168,0	8,6	191,3	1	[b]	[b]	[b]	[b]	[b]	192	163,1	8,6	191,3	1
RIO25E_143	6	0	0	5,3	1	6	0,2	0	4,7	1	6	0	0	5,3	1
RIO25G_144	36	0	0	35,3	1	36 (1)	7207,4	7,2	30	7349	36	0	0	35,3	1
RIO25N_145	144	0,6	0,4	143,0	1	[a]	724,5	715,5	[b]	1	144	1,0	0,4	143,0	1
STE1A_146	5	0	0	4,7	1	5	12,5	0	4,1	181	5	0	0	4,7	1
STE1R_147	3	0	0	2,1	1	3	895,5	0	1,5	411810	3	0	0	2,1	1

De um modo geral, resumindo os resultados obtidos para as instâncias A e B, o modelo proposto é capaz de resolver quase instantaneamente instâncias reais com menos de 20 tipos de itens e instâncias reais com mais de 40 tipos de itens em poucos minutos. Poucas instâncias fornecidas pelas duas empresas de mobiliário têm mais de 40 tipos de itens. Para todas as instâncias, exceto em duas, a solução ótima ou soluções com um gap de uma ou duas placas (significa que a solução é ótima ou a solução ótima usa menos uma ou duas placas) foram obtidas em tempo aceitável. Para o problema de dois-estágios não-exato, o modelo proposto não conseguiu alcançar a solução ótima em 5 das 168 instâncias, enquanto o modelo proposto por Lodi et al. (2004) não foi capaz de encontrar a solução ótima em 121 de um total de 168 instâncias. Para o problema de três-estágios exato, o modelo proposto não obteve a solução ótima em 6 das 168 instâncias, enquanto o modelo proposto por Puchinger & Raidl (2007) não atingiu a solução ótima em 121 das 168 instâncias.

[a] Cplex 11.0 retornou "falta de memória".

[b] Memória insuficiente para construir o modelo.

Instâncias adaptadas da literatura

Para o conjunto das instâncias C (tabela 3.14 e tabela 3.15), os modelos propostos obtiveram a solução ótima em todos os casos, exceto na instância CHL6 para o problema 3E. A mesma instância para o mesmo problema também não foi resolvida pelo modelo de Puchinger & Raidl (2007).

Existem também dois outros casos para os quais os modelos da literatura não conseguiram encontrar a solução ótima (Hchl7s, 2NE e CW1,3E) devido a memória insuficiente. O motivo que justifica o facto destas instâncias serem mais difíceis de resolver que as restantes do conjunto não é obvio. Possivelmente, a qualidade dos limites inferiores e superiores (utilizados na partição e avaliação sucessivas) para estas instâncias seja pior do que em outras instâncias, devido a aspectos geométricos.

Para o problema mais fácil (2E) o modelo proposto resolveu o conjunto de instâncias C quase instantaneamente, o tempo computacional máximo foi de apenas 2,9 segundos para a instância CHL7.

Tal como o esperado os tempos computacionais aumentaram para o problema três-estágios não-exato quando comparado com os outros tipos de problemas (2E, 2NE, 3E), mas o modelo proposto ainda foi capaz de resolver todas as instâncias otimamente.

Tabela 3.14: Resultados para o conjunto de instâncias C para o problema de dois-estágios.

Instância	$2E$					$2NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
A1	27	0	0	26,0	1	23	0	0	22,5	1	23	0	0	14,9	1
A2	15	0	0	12,9	1	12	0,1	0,1	11,6	1	12	0,3	0	9,6	1
A3	10	0,1	0,1	7,4	1	8	0,2	0,1	7,1	1	8	0,7	0	6,2	1

Tabela 3.14 Resultados para o conjunto de instâncias C para o problema de dois-estágios *(continuação da página anterior).*

Instância	$2E$					$2NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
A4	8	0,1	0	4,4	1	5	0,9	0,2	4,1	41	5	0,3	0	3,8	1
A5	8	0,3	0,1	4,5	21	5	0,9	0,4	4,0	1	5	0,5	0	3,7	1
CHL1	11	0,2	0,1	5,8	1	6	54,9	1,2	5,3	471	6	5,8	0	5,0	478
CHL2	4	0	0	2,3	1	3	0,1	0	2,2	1	3	0	0	2,1	1
CHL5	5	0	0	3,6	1	4	0	0	3,3	1	4	0	0	2,6	1
CHL6	9	2,5	0,2	5,2	488	6	4,6	0,9	5,0	1	6	1,3	0	4,8	1
CHL7	9	2,9	0,3	5,6	532	6	15,0	1,0	5,2	101	6	1,4	0	5,1	1
CU1	15	0,1	0	12,0	21	12	0,4	0,2	11,2	1	12	2,4	0	10,4	11
CU2	20	0,2	0,1	18,0	1	15	0,3	0,2	14,0	1	15	1,3	0	12,9	1
CW1	13	0,3	0,1	10,3	31	10	0,3	0,2	9,1	1	10	0,8	0	8,5	1
CW2	17	0,2	0,1	13,9	1	12	4,2	0,2	11,8	833	12	2,3	0	10,8	190
CW3	22	0,7	0,2	18,7	41	16	3,0	0,3	15,3	190	16	6,3	0	14,3	201
Hchl2	9	1,5	0,3	5,7	44	6	88,4	1,1	5,3	674	6	18,1	0	5,2	784
Hchl3s	4	0,1	0,1	2,8	1	3	0,7	0,1	2,7	61	3	0,3	0	2,6	1
Hchl4s	3	0,1	0	1,8	1	2	0,7	0,1	1,7	46	2	0,9	0	1,7	162
Hchl6s	7	0,4	0,2	4,7	1	5	6,7	0,6	4,4	111	5	2,5	0	4,2	131
Hchl7s	11	2,0	0,5	6,8	31	7	75,2	3,7	6,4	71	8(1)[a]	3033,7	0	6,3	74318
Hchl8s	3	0	0	1,1	1	2	0,1	0,1	1,1	1	2	0	0	1,2	1
Hchl9	14	0,3	0,1	10,4	1	10	5,3	0,4	9,6	101	10	87,2	0	9,1	3695
HH	2	0	0	1,2	1	2	0	0	1,2	1	2	0	0	1,2	1
OF1	5	0	0	3,1	1	4	0	0	3,1	1	4	0	0	2,9	1
OF2	6	0	0	4,3	1	5	0	0	3,9	1	5	1,1	0	3,2	898
STS2	17	0,2	0,1	15,0	1	12	1,7	0,2	11,6	129	12	165,9	0	10,8	7483
STS4	6	0,1	0,1	5,0	1	5	1,2	0,2	4,6	21	5	5,5	0	4,4	509
W	31	0	0	28,8	1	24	0,1	0,1	23,4	1	24	0,2	0	15,9	1
2	3	0,1	0	1,9	1	2	0,3	0	1,8	27	2	0,1	0	1,6	1
3	24	0,1	0	23,2	1	23	0,1	0	22,5	1	23	0,1	0	15,9	1

Tabela 3.15: Resultados para o conjunto de instâncias C para o problema de três-estágios.

Instância	$3E$										$3NE$				
	Modelo proposto					Modelo proposto					Puchinger et al. (2007)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
A1	23	0,1	0	22,5	1	23	0,1	0	14,8	1	23	0,1	0,1	22,5	1
A2	12	0,1	0,1	11,5	1	12	0,4	0	9,6	1	12	0,1	0,1	11,5	1
A3	8	0,2	0,2	7,1	1	8	8,0	0	6,2	537	8	0,2	0,2	7,0	1

[a]Cplex 11.0 retornou ”falta de memória”.

Tabela 3.15 Resultados para o conjunto de instâncias C para o problema de três-estágios *(continuação da página anterior).*

Instância	$3E$										$3NE$				
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
A4	5	1,1	0,6	4,1	1	5	0,1	0	3,8	1	5	2,4	0,6	4,1	21
A5	4	44,7	0,7	3,8	1417	4	81,5	0	3,6	4952	4	100,7	0,7	3,8	3118
CHL1	6	25,5	2,0	5,0	71	6	7,3	0	4,9	181	6	21,9	2,0	5,0	81
CHL2	3	0,1	0,1	2,2	1	3	0	0	2,1	1	3	0,1	0	2,2	1
CHL5	3	0	0	2,7	1	3	0	0	2,5	1	3	0,1	0	2,7	1
CHL6	6 (1)	7201,7	1,7	4,9	209354	6(1)[a]	5136,1	0	4,7	106293	5	489,1	1,7	4,9	7397
CHL7	6	27,8	1,8	5,2	81	6	1,5	0	5,0	1	6	27,1	1,8	5,2	121
CU1	12	1,1	0,3	11,2	51	12	5,3	0,1	10,4	131	12	4,0	0,3	11,2	471
CU2	14	6,8	0,3	13,9	920	14	4,9	0	12,8	161	14	1,6	0,3	13,9	108
CW1	10	0,6	0,3	9,0	1	10(1)[a]	4992,0	0	8,4	144431	10	2,7	0,3	8,9	616
CW2	12	1,4	0,3	11,7	150	12	0,7	0	10,7	1	12	1,0	0,3	11,6	31
CW3	16	1,1	0,4	15,1	1	16	4,7	0,1	14,2	71	16	1,2	0,5	15,0	1
Hchl2	6	39,1	1,9	5,3	201	6	12,7	0	5,1	483	6	44,3	1,9	5,3	201
Hchl3s	3	2,6	0,2	2,7	171	3	4,0	0	2,6	241	3	2,2	0,3	2,7	141
Hchl4s	2	1,2	0,2	1,7	31	2	0,3	0	1,7	1	2	2,5	0,3	1,7	111
Hchl6s	5	12,4	1,1	4,3	71	5	2,1	0	4,2	101	5	3,5	1,1	4,3	1
Hchl7s	7	2676,8	5,8	6,4	2789	7	28,8	0	6,3	488	7	2905,6	5,9	6,4	2364
Hchl8s	2	0,3	0,2	1,0	1	2	0	0	1,0	1	2	0,3	0,2	1,0	1
Hchl9	10	9,4	0,8	9,5	101	10	19,7	0	9,1	497	10	19,6	0,9	9,5	311
HH	2	0	0	1,2	1	2	0	0	1,1	1	2	0	0	1,2	1
OF1	4	0,1	0,1	3,0	1	4	0	0	2,7	1	4	0,1	0,1	3,0	1
OF2	4	0,1	0	3,7	4	4	0,8	0	3,1	448	4	0,1	0	3,7	27
STS2	12	1,5	0,4	11,6	61	12	318,7	0	10,8	14079	12	6,9	0,4	11,5	471
STS4	5	2,3	0,5	4,6	51	5	2,9	0	4,3	161	5	2,2	0,5	4,6	51
W	24	0,1	0,1	23,4	1	24	0,4	0	15,9	1	24	0,2	0,1	23,4	1
2	2	0,1	0,1	1,7	1	2	0	0	1,6	1	2	0,1	0,1	1,6	1
3	23	0,1	0,1	22,5	1	23	0,1	0	15,9	1	23	0,1	0,1	22,5	1

O conjunto de instâncias D (tabela 3.16 e tabela 3.17) é o conjunto mais difícil porque cada placa pode acomodar um grande número de itens (para o problema 2NE, as soluções obtidas pelo modelo proposto têm cerca de 20 itens por placa) e as instâncias têm um número considerável de tipos de itens (entre 25 a 58 com uma média de 40), assim como um número considerável de itens (entre 119 a 325 com uma média de 202). O número de tipos de itens e o número de itens

[a]Cplex 11.0 retornou "falta de memória".

são fatores cruciais para o tamanho do modelo proposto e para os modelos da literatura, respetivamente.

No problema dois-estágios exato, o modelo proposto forneceu a solução ótima para todas as instâncias e 65% das instâncias foram resolvidas em menos de 30 segundos.

Para o problema de dois-estágios não-exato, os resultados obtidos pelo modelo proposto são melhores do que os do modelo descrito em Lodi et al. (2004) (uma solução melhor foi obtida em 9 das 20 instâncias, uma solução com o mesmo valor mas com um gap mais pequeno foi obtido em 3 instâncias e para 8 instâncias, o valor e o gap das soluções foram os mesmos). Para o problema de três-estágios exato, os resultados do modelo proposto são ligeiramente piores que os do modelo proposto em Puchinger & Raidl (2007) (uma solução melhor foi obtida em 4 de 20 instâncias, mas uma solução pior foi encontrada em seis instâncias). A melhoria do limite inferior do modelo proposto com respeito aos modelos da literatura é mais pequena (3% para 2NE e 2% para 3E) do que a melhoria para outros conjuntos de instâncias.

Para o problema de três-estágios não-exato, apenas 5 em 20 instâncias foram resolvidas otimamente pelo modelo proposto e uma solução admissível não foi encontrada (em 7200 segundos) para duas instâncias. Para as restantes 13 instâncias, embora a otimalidade não tenha sido comprovada, as soluções de facto ótimas usam apenas mais uma ou duas placas que a solução obtida.

Tabela 3.16: Resultados para o conjunto de instâncias D para o problema de dois-estágios.

Instância	$2E$					$2NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
ATP30	12	21,7	0,5	9,2	529	9	206,1	14,8	8,2	1	9	278,3	0,2	7,8	2898
ATP31	19	218,3	3,5	15,1	131	15 (1)	7257,8	57,3	13,3	236	15 (1)	7200,4	0,3	13,1	22759
ATP32	16	4,7	0,6	13,0	51	13	893,7	11,4	12,0	261	14 (2)	7203,1	0,3	11,7	23135

Tabela 3.16 Resultados para o conjunto de instâncias D para o problema de dois-estágios *(continuação da página anterior).*

Instância	$2E$					$2NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
ATP33	18	131,5	2,8	13,6	101	13 (1)	7210,6	10,4	11,9	2098	14(2)[a]	6810,4	0,3	11,5	27531
ATP34	9	82,0	0,9	5,9	543	6	303,6	5,0	5,2	201	6	45,3	0,1	5,0	764
ATP35	10	3,8	1,3	7,9	1	8	415,2	8,0	7,2	101	8	73,6	0,1	6,9	843
ATP36	11	5,0	0,4	8,0	121	8	220,2	4,9	7,2	271	8	1748,3	0,1	7,0	21683
ATP37	16	78,1	2,5	11,9	81	12	2496,2	20,1	11,0	223	13 (2)	7570,3	0,2	10,6	28313
ATP38	15	413,5	1,0	11,4	31987	11	149,6	13,9	10,1	1	11(1)[a]	6196,8	0,2	9,7	30270
ATP39	16	6,9	0,9	12,5	21	12 (1)	7206,8	6,7	10,9	33271	12(1)[a]	4703,5	0,1	10,6	38135
ATP40	20	25,4	0,7	16,0	495	16 (1)	7232,1	31,9	14,7	629	17 (2)	7200,6	0,5	14,5	13345
ATP41	16	7,2	0,8	12,9	61	12	29,2	6,2	11,2	1	13(1)[a]	4201,5	0,1	10,9	15500
ATP42	21	3138,6	1,3	16,4	101438	16 (1)	7212,7	12,5	14,5	993	17 (2)	7201,0	0,6	14,2	22865
ATP43	18	139,7	3,2	13,4	111	14 (1)	7224,8	24,7	12,0	519	14(2)[a]	6425,3	0,3	11,7	31782
ATP44	14	14,2	1,3	9,3	21	9	3583,9	9,3	8,3	728	10(1)[a]	5281,8	0,1	8,1	40665
ATP45	11	3,0	1,1	8,1	1	8	1724,8	5,3	7,4	1254	9(1)[a]	3049,8	0,1	7,2	31989
ATP46	16	22,9	1,4	12,5	101	12 (1)	7216,1	16,0	10,8	844	12(1)[a]	4185,5	0,2	10,5	30820
ATP47	18	5,9	1,4	13,5	1	13	778,2	14,7	12,0	201	13(1)[a]	4603,6	0,2	11,7	37151
ATP48	11	11,8	0,6	8,6	201	9 (1)	7218,7	18,6	7,7	1046	9(1)[a]	4494,3	0,1	7,4	55015
ATP49	8	6,9	0,8	5,3	21	5	4271,2	7,6	4,8	1823	6(1)[a]	2169,3	0	4,7	34645

Tabela 3.17: Resultados para o conjunto de instâncias D para o problema de três-estágios.

Instância	$3E$					$3NE$									
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
ATP30	9 (1)	7232,5	32,3	8,0	530	9 (1)	7200,7	0,6	7,8	23516	-	7232,6	32,5	7,9	603
ATP31	-	7335,5	135,2	13,2	141	15 (1)	7201,7	1,5	13,0	14504	15 (1)	7345,5	145,0	13,2	48
ATP32	14 (2)	7228,4	28,1	11,9	596	13 (1)	7201,4	1,3	11,7	10993	14 (2)	7229,2	28,9	11,8	102
ATP33	13 (1)	7217,0	16,9	11,8	1428	14 (2)	7201,1	0,9	11,5	16904	13 (1)	7219,0	18,9	11,7	1091
ATP34	6	306,7	15,5	5,1	1	6	25,0	0,1	5,0	401	6	207,3	16,1	5,1	1
ATP35	8	2161,6	22,5	7,1	301	8	160,8	0,3	6,9	519	8	1229,0	23,9	7,1	201
ATP36	8	171,9	11,5	7,1	1	8	2834,7	0,3	7,0	22119	8	570,4	11,5	7,1	171
ATP37	15 (4)	7239,1	39,0	10,7	372	12 (1)	7201,0	0,9	10,6	15546	12 (1)	7241,7	41,4	10,7	325
ATP38	12 (2)	7239,9	39,7	9,9	296	12 (2)	7200,8	0,6	9,7	17295	11 (1)	7242,2	42,0	9,8	284
ATP39	12 (1)	7215,0	14,8	10,8	1647	12 (1)	7200,4	0,3	10,6	17682	12 (1)	7215,3	15,2	10,8	888
ATP40	-	7280,5	80,2	14,6	31	16 (1)	7202,2	2,0	14,4	8714	-	7281,1	80,7	14,6	221
ATP41	12	597,1	27,5	11,1	1	13 (2)	7200,5	0,4	10,9	29314	12	2198,7	28,3	11,1	131
ATP42	17 (2)	7221,2	21,0	14,4	806	16 (1)	7202,9	2,8	14,2	8393	16 (1)	7222,9	22,8	14,4	842

[a]Cplex 11.0 retornou "falta de memória".

Tabela 3.17 Resultados para o conjunto de instâncias D para o problema de três-estágios *(continuação da página anterior).*

Instância	$3E$										$3NE$				
	Modelo proposto					Modelo proposto					Lodi et al. (2004)				
	z	t	c	lb	nbb	z	t	c	lb	nbb	z	t	c	lb	nbb
ATP43	15 (3)	7249,4	49,2	11,9	385	14 (2)	7201,6	1,5	11,7	10541	14 (2)	7256,6	56,4	11,8	320
ATP44	9	5769,3	15,3	8,2	1085	10(1)[a]	6864,5	0,7	8,1	30352	10 (1)	7216,5	16,4	8,2	1058
ATP45	8	852,5	6,6	7,3	704	9 (1)	7200,5	0,3	7,1	37811	8	3365,8	6,8	7,3	2312
ATP46	12 (1)	7235,5	35,0	10,7	425	12 (1)	7200,7	0,6	10,5	22983	12 (1)	7236,3	36,2	10,7	580
ATP47	14 (2)	7228,2	28,0	11,9	768	14 (2)	7200,7	0,6	11,7	21918	13 (1)	7229,7	29,2	11,9	216
ATP48	9 (1)	7237,3	37,1	7,6	335	9 (1)	7203,6	0,4	7,4	33101	9 (1)	7238,1	37,8	7,5	461
ATP49	6 (1)	7216,9	16,8	4,8	734	6(1)[a]	4202,4	0,1	4,7	52255	6 (1)	7217,3	17,0	4,8	519

3.4.3 Comparação com modelos de geração de colunas

Foram implementados dois modelos de geração de colunas em que a principal diferença entre ambos reside nos modelos utilizados para a resolução dos subproblemas, isto é nos modelos utilizados para a criação de padrões de corte atrativos para o problema mestre.

O primeiro modelo implementado foi a abordagem clássica de Gilmore & Gomory (1965), cuja formulação do problema mestre é apresentada em (2.1) a (2.3). Considerou-se o problema mestre restrito com tantas colunas quantos o número de itens, sendo que cada padrão foi construído com o número máximo de itens que pode ser alocado a uma placa. Para a resolução dos subproblemas utilizou-se uma sequência de dois tipos de problemas de mochila também propostos por Gilmore & Gomory (1965), que foram anteriormente introduzidos de (2.4) a (2.10).

O segundo modelo de geração de colunas implementado utiliza a mesma formulação que Gilmore & Gomory (1965) para o problema mestre, no entanto para

[a]Cplex 11.0 retornou "falta de memória".

a resolução do subproblema resolve uma sequência de problemas a uma dimensão através de formulações baseadas em Dyckhoff (1981).

Na formulação de Dyckhoff (1981) cada variável de decisão corresponde a uma única operação de corte executada num único objeto (no caso do problema unidimensional os objetos podem ser por exemplo rolos ou barras). Com a realização da operação de corte o objeto é cortado em duas peças e pelo menos uma corresponde ao tamanho de um item procurado. O corte pode ser efetuado em objetos grandes ou em peças que resultam de operações de corte anteriores. O modelo tem por base a enumeração de todas as operações de corte possíveis e de todas as peças resultantes de cortes que podem posteriormente ser cortadas.

De modo a fornecer ao problema mestre padrões de corte do tipo dois-estágios não-exato é resolvida uma sequência de $n + 1$ problemas unidimensionais, nos primeiros n problemas é resolvido o modelo de programação inteira (3.9)-(3.14). Neste modelo uma tira horizontal com altura h_i é iniciada com o item i e completada com itens com altura inferior ou igual a h_i, sendo que se pretende selecionar os itens mais valiosos, os coeficientes (π_i, $i = 1, ..., n$) da função objetivo são os valores das variáveis duais associadas às restrições de procura dos itens no problema mestre (2.1) - (2.3). As variáveis x_{lj} significam que o item l foi cortado da peça residual j e x_{i0} significa que o item i inicia a tira, nas restrições (3.10) pretende-se que seja utilizado na tira o maior número de possível itens, nas restrições (3.11) garante-se que só serão efetuados cortes em peças residuais existentes e por último a restrição (3.14) garante que a tira é iniciada pelo item i.

Na descrição do modelo assumiu-se que os cortes e as peças residuais eram conhecidos, no entanto para a enumeração dos cortes e tamanho das peças residuais é proposto o algoritmo 2, no final a lista R conterá todas as peças residuais possíveis e a lista C todos os cortes possíveis.

Como descrito, o primeiro tipo de problemas é resolvido para cada item i

$i = 1, ..., n$, sendo calculado o respetivo Z_i^* valor máximo obtido para a tira iniciada pelo item i, criando n tiras horizontais inicializadas pelos diferentes tipos de itens.

Uma vez geradas as tiras horizontais, é necessário escolher as que poderão construir o padrão de corte com valor mais atrativo para o problema mestre. Este problema é resolvido de forma semelhante à criação de tiras horizontais, o modelo proposto é apresentado de (3.15) a (3.19), a principal diferença entre este modelo e o anterior reside na restrição (3.18) na qual se decide qual a tira horizontal que inicializa a placa. O algoritmo 2 pode também ser utilizado para a geração dos cortes e das peças residuais, sendo que a largura é substituída pela altura.

$$\text{Maximizar } \mathrm{Z_i^*} = \sum_{l=1}^{n}\sum_{j=0}^{m} \pi_l x_{lj} \tag{3.9}$$

$$\text{Sujeito a:} \quad \sum_{j=0}^{m} x_{lj} \geq 0, \qquad \forall l \in I^* \tag{3.10}$$

$$\sum_{j=0}^{m}\sum_{l=1}^{n} a_{ijk} x_{lj} + \sum_{l=1}^{n} x_{lk} \geq 0 \qquad k = 1, ..., m, \tag{3.11}$$

$$x_{i0} = 1, \tag{3.12}$$

$$I^* = \{h_l : hl \leq h_i, i = 1, ..., n\} \tag{3.13}$$

$$x_{lj} \geq 0 \text{ e inteiro}, \qquad l = 1, ..., n, j = 0, ..., m. \tag{3.14}$$

$$\text{Maximizar} \quad \sum_{i=1}^{n}\sum_{j=0}^{m} Z_i^* y_{ij} \tag{3.15}$$

$$\text{Sujeito a:} \quad \sum_{j=0}^{m} y_{ij} \geq 0, \qquad i = 1, ..., n \tag{3.16}$$

$$\sum_{j=0}^{m}\sum_{i=1}^{n} a_{ijk} y_{ij} + \sum_{i=1}^{n} y_{ik} \geq 0 \qquad k = 1, ..., m, \tag{3.17}$$

$$\sum_{i=1}^{n} y_{i0} = 1, \tag{3.18}$$

$$y_{ij} \geq 0 \text{ e inteiro}, \qquad i = 1, ..., n, j = 0, ..., m. \tag{3.19}$$

Algoritmo 2: Algoritmo para gerar cortes e objetos residuais.

início
- Inicializar R com o conjunto vazio;
- Inicializar C com n cortes correspondendo ao corte do item do tipo i de uma tira com altura h_i e largura W (largura da placa inicial);
- Marcar todos os cortes C como não analisados;
- **enquanto** *não forem analisados todos os cortes de* C **faça**
  - Seja c um corte não analisado de C associado com o tipo de item i e uma peça residual ou objeto inicial;
  - Marcar c como analisado;
  - Adicionar a nova peça residual com altura h_i e largura $W_l - w_i$ à lista R se for maior ou igual à menor largura dos itens e se ainda não pertencer à lista;
  - **para todo** *itens do tipo* $j : h_j \leq h_i$ **faça**
    - **se** *o item do tipo i pode ser cortado de uma peça residual* **então**
      - Adicione o corte associado à peça residual e ao item do tipo i a C e marque-o como não analisado
    - **fim**
  - **fim**
- **fim**

fim

É importante salientar que os modelos de geração de colunas apresentados apenas resolvem a relaxação linear e não o problema inteiro. De modo a obter uma solução inteira em ambos os modelos de geração de colunas resolve-se um problema de programação inteira com todas as colunas presentes no problema mestre restrito, obtendo-se assim heuristicamente uma solução inteira.

Tal como nos testes realizados com os modelos de afetação foi definido um limite de 7200 segundos para o *solver* Cplex para os modelos de geração de colu-

nas.

Salienta-se que os padrões de corte obtidos com os modelos de geração de colunas são do tipo dois-estágios não-exato (2NE) e serão portanto comparados com as soluções obtidas pelo modelo de programação inteira para o problema referido.

O modelo de geração de colunas em que os subproblemas são resolvidos através de formulações do problema de mochila de Gilmore & Gomory (1965) será denominado de "modelo de geração de colunas 1" enquanto que o modelo em que os subproblemas têm por base a formulação de Dyckhoff (1981) para o problema unidimensional será denominado de "modelo de geração de colunas 2".

Para o modelo de programação inteira proposto é apresentada a solução ótima e também o tempo necessário para a obter, incluindo o tempo de construção do modelo. Para os modelos de geração de colunas é apresentado também o valor da solução obtida (colunas Z). Relembra-se que a heurística consiste em resolver um problema de programação inteira com todas as colunas inseridas no problema mestre restrito ao longo do processo de geração de colunas para o cálculo da relaxação linear.

Nas colunas T_{H1} e T_{H2} é apresentado o tempo computacional necessário para obter a solução inteira com a heurística com o modelo de geração de colunas 1 e 2 respetivamente, este valor inclui o tempo necessário para obter a relaxação linear. O número de colunas adicionadas ao problema mestre restrito é apresentado nas colunas N_{col}.

Instâncias reais

Na tabela 3.18 são apresentados os resultados computacionais para as instâncias da empresa A.

A heurística com o modelo de geração de colunas 1 chegou ao ótimo em apenas

19 instâncias, enquanto que com o modelo de geração de colunas 2 em 20, das soluções que atingiram o ótimo em apenas 7 instâncias a heurística foi mais rápida do que o modelo de programação inteira.

O modelo de geração de colunas 1 inseriu no total das instâncias mais colunas no problema mestre, tendo inserido em média mais 34 colunas.

Tabela 3.18: Resultados para o conjunto de instâncias da empresa A.

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
AP-9_1	61	376	63	85	58	62	11,6	72
AP-9_10	65	9,7	65	6,8	47	65	3,3	45
AP-9_11	58	18,8	58	16,9	65	58	5,8	59
AP-9_12	27	1,3	27	0,5	19	27	0,5	17
AP-9_13	28	25,5	28	11	77	28	9,5	73
AP-9_14	3	0	3	0	0	3	0	1
AP-9_15	14	0,2	14	0,5	16	14	0,6	16
AP-9_2	3	0,1	3	0	6	3	0	6
AP-9_3	46	81,7	47	24,6	51	47	5,6	55
AP-9_4	14	0	14	0	1	14	0	2
AP-9_5	14	0,9	14	0,2	9	15	0,2	9
AP-9_6	67	432,3	69	1161,4	105	68	59,8	99
AP-9_7	39	0,9	40	0,4	17	40	0,8	20
AP-9_8	83	53,4	83	32,4	86	83	19,6	106
AP-9_9	5	0	5	0	4	5	0,1	5
AP-9-3MM-4MM_1	4	0	4	0	2	4	0,6	2
AP-9-3MM-4MM_2	36	0	36	0	3	36	0,7	3
AP-9-3MM-4MM_3	8	0	8	0	2	8	0	2
AP-9-3MM-4MM_4	3	0	3	0	2	3	0	2
AP-9-3MM-4MM_5	13	0,2	14	0,3	13	14	3,2	18
AP-9-3MM-4MM_6	2	0	3	0	1	3	0	2
AP-9-3MM-4MM_7	14	0	14	0	4	14	0,1	6
AP-9-3MM-4MM_8	2	0	2	0	0	2	0	1
FA+AA-9_1	35	2275,1	-	0	1969	36	1863,9	504
FA+AA-9_10	8	46,9	9	34,2	146	9	43,5	173
FA+AA-9_11	27	485,5	32	1984,9	369	30	293,7	368
FA+AA-9_12	19	639,2	22	399,3	372	21	196,3	370
FA+AA-9_13	39	7363,5	-	-	-	39	4494,4	615
FA+AA-9_14	6	3,6	8	11,6	76	7	7,5	73
FA+AA-9_15	17	330,4	19	719,5	287	20	169,2	298
FA+AA-9_2	18	3380,7	20	1261,1	436	19	382,1	451
FA+AA-9_3	42	7278,6	44	3950,3	466	43	639,5	492
FA+AA-9_4	8	36,2	10	30,3	132	10	27,8	117
FA+AA-9_5	20	292	23	195,6	146	22	41,9	156
FA+AA-9_6	20	4333	22	4618,2	467	22	3221,7	474
FA+AA-9_7	12	829,9	14	337,2	237	13	109,4	239
FA+AA-9_8	28	4197	29	2546,4	407	29	664,9	403
FA+AA-9_9	4	74,6	7	12,2	76	5	15,2	98

Tabela 3.18 Resultados para o conjunto de instâncias da empresa *A*

(continuação da página anterior).

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
FA+AP-9-10MM_1	9	0,7	11	0,5	17	11	0,7	17
FA+AP-9-10MM_2	5	0,1	6	0,3	18	5	0,6	20
FA+AP-9-10MM_3	23	41,8	25	13,9	98	24	17,1	112
FA+AP-9-10MM_4	4	0,7	5	0,7	27	5	0,9	25
TRAS-BC_1	15	17,7	17	30,8	118	16	18,2	120
TRAS-BC_2	17	7201	20	19,9	95	19	15,2	93
TRAS-BC_3	19	2,1	19	33,8	52	19	3,5	49
TRAS-BC_4	8	0	9	0,1	5	9	0,1	7
TRAS-BC_5	7	0,1	7	0,3	15	7	0,3	11

Os resultados computacionais referentes à empresa B são apresentados na tabela 3.19. Os valores da relaxação linear são iguais para todos os modelos e para todas as instâncias e o tempo computacional foi em 93% dos testes inferior a um minuto. Para este conjunto de instâncias a heurística alcançou o ótimo em apenas 55% e 53% das instâncias nos modelos de geração de colunas 1 e 2, respetivamente e em nenhuma das vezes que atingiu o valor ótimo foi mais rápida que o modelo de programação inteira. Tal como no conjunto de instâncias da empresa A, o modelo de geração de colunas 1 inseriu mais colunas no problema mestre restrito do que o modelo de geração de colunas 2.

Tabela 3.19: Resultados para o conjunto de instâncias da empresa *B*.

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
03G18_1	76	0,2	77	0,4	10	76	0,4	7
03J18_2	171	0,1	172	0,8	19	173	0,5	11
03X22_3	110	0	111	0,1	4	111	0	1
10C30_4	21	0	22	0,1	6	22	0	4
10M30_5	11	0	11	0	5	11	0	3
10W30_6	10	0	10	0	3	10	0	1
114M18_7	537	0	537	0	4	537	0	3
114UU18_9	334	0	336	0,1	6	335	0	3

Tabela 3.19 Resultados para o conjunto de instâncias da empresa B *(continuação da página anterior).*

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
12D16_10	85	10,3	85	30,4	84	85	202,9	81
12F18_11	78	3,8	81	20,1	44	81	229	45
12M18_12	271	8,1	272	850,6	96	272	996,3	94
146O18_17	465	0	466	0,4	13	466	0,1	8
146OO18_18	230	0,2	230	0,9	20	230	1,1	19
15A30_19	35	0	35	0,1	6	36	0,1	5
15B30_20	16	0	16	0,1	10	16	0,1	9
15G30_21	24	0	25	0,3	9	25	0,2	10
15J30_22	12	0	12	0	2	12	0	0
15M30_23	14	0	14	0	3	14	0	1
15N30_24	47	0,1	47	1,2	24	48	0,8	21
15X18_25	95	0	95	0,2	6	95	0,1	6
15X22_26	229	0,1	229	0,1	4	229	0,1	4
15Y30_27	17	0	17	0,2	10	17	0,1	9
16B18_28	85	0,2	86	1,6	24	85	1,5	24
174N25_31	9	0	10	0	2	10	0	1
18SC18_36	47	0	48	0,1	5	48	0	3
214W18_42	128	0,1	129	0,6	12	129	0,4	9
214Y18_43	221	0	221	0,2	12	221	0,1	8
22A18_48	38	0,2	38	0,2	9	38	0,3	7
22G18_49	19	0	20	0,1	5	20	0,1	4
22X18_50	57	2,2	59	3	28	59	8,5	24
23A25_51	30	0	31	0,1	5	31	0	3
23B25_52	192	1,2	192	19,4	84	192	19,1	83
23C25_53	52	0	52	0,2	10	52	0,1	8
23N25_54	83	0,4	84	1	17	84	0,9	20
23NATO18_55	96	0	96	0	4	96	0	3
23NATO25_56	96	0	98	0	4	98	0	0
23R25_57	53	0	53	0,1	9	53	0,1	7
26K19_58	35	0	36	0	2	36	0	0
26R16_59	35	0	35	0	2	35	0	1
28C18_60	19	0	20	0,2	8	20	0	3
28C25_61	17	0	18	0,2	7	18	0,1	5
28C30_62	16	0	17	0,2	9	17	0,1	4
28C38_63	6	0	6	0	2	6	0	0
28CC18_64	24	0,4	25	1,5	25	25	2,7	31
28R18_65	11	0	11	0	2	11	0	0
28R25_66	28	0	28	0,1	7	28	0,1	6
28R30_67	8	0	8	0	4	8	0	1
28T18_68	14	0	15	0,3	10	15	0,1	8
30A63_69	21	0	21	0	3	21	0	3
30R1_70	71	0	71	0	1	71	0	0
35B22_71	63	0	64	0,2	9	63	0,1	8
37A18_73	120	0	120	0	1	120	0	0
37W18_74	3	0	5	0	4	4	0	4
38SC18_75	13	0	14	0,2	9	14	0,1	7
39H18_76	51	0,3	51	0,9	21	51	2,3	17
39N18_77	214	1,1	215	7,1	60	215	24,7	55
39Q18_78	44	1	45	13	59	45	31,3	51

Tabela 3.19 Resultados para o conjunto de instâncias da empresa *B* *(continuação da página anterior).*

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
39R18_79	20	0,1	20	0,3	10	20	0,2	8
40A63_80	11	0	11	0	2	11	0	1
40C1_81	13	0	13	0,1	5	13	0,1	5
40C2_82	6	0	6	0	2	6	0	1
40R1_83	18	0	19	0,1	6	18	0,1	5
42B18_84	38	0,1	39	0,3	10	39	0,2	9
42B22_85	20	0,3	20	1,2	26	20	0,7	18
42C18_86	172	1,3	172	11,2	40	173	118	37
42C22_87	41	0,1	41	0,7	18	41	0,4	17
42F18_88	262	11,8	262	20,8	55	263	105,3	52
42F22_89	49	0,7	50	3,4	42	50	3,3	38
42K18_90	20	0	21	0	3	21	0	3
42N18_91	117	0,9	117	3,4	34	117	6,1	27
42N22_92	66	0,2	66	0,9	19	67	0,7	19
42O18_93	34	0,1	34	0,6	18	35	0,7	18
42R18_94	19	0,1	19	0,2	9	19	0,2	10
43L16_95	309	0,1	310	0,1	6	310	0,1	4
451W18_97	126	0,1	127	0,4	12	127	0,4	9
451Y18_98	221	0	221	0,2	12	221	0,1	8
45B16_99	208	1,1	208	4	47	208	4,4	39
45C16_100	73	0	73	0,3	12	73	0,3	16
45K16_101	50	0,1	50	0,5	16	50	0,4	18
45R16_102	57	0,1	58	0,4	11	58	0,4	12
58F18_103	425	0,4	426	1,3	21	426	2,1	22
58G18_104	105	0,2	106	0,6	20	106	0,6	21
58M18_105	97	0,3	98	1,1	20	99	0,8	14
58O18_106	103	0,3	103	0,5	13	105	0,5	14
64M18_107	106	0,4	106	0,5	10	106	1	8
68M18_108	106	0,4	106	0,5	10	106	1	8
68W18_109	128	0,1	129	0,6	13	129	0,8	15
68Y18_110	226	0	226	0,2	14	226	0,1	8
69W18_111	353	0,2	353	0,9	18	353	1,1	21
69Y18_112	841	0	842	0,3	11	841	0,2	12
69Z18_113	317	0,1	318	1,2	12	318	0,2	9
705MM18_114	176	0,3	176	0,4	10	176	0,5	7
70M18_116	201	0,4	201	0,4	12	201	1,3	9
70Z18_117	12	0	12	0,1	5	12	0	2
71R18_118	60	0	60	0	2	60	0	3
71W18_119	124	0	125	0,1	5	124	0,1	4
76R25_121	60	0	61	0	2	61	0	1
86R25_122	68	0,2	69	0,3	10	69	0,4	7
94H18_123	1095	0	1096	0	4	1096	0	1
ANT39C_125	59	1,1	60	2,4	25	60	5,6	22
ANT39R_126	59	1,7	60	12,3	34	60	29,1	29
ANT41C_127	46	0,5	49	1,3	20	50	3,5	13
ARKEN18P_129	41	0,1	41	0,3	11	41	0,2	10
ARKEN25F_130	3	0	3	0	1	3	0	0
ARKEN25P_131	34	0	34	0	1	34	0	0
CARLET_132	71	0,4	71	2,6	36	71	1,9	29

Tabela 3.19 Resultados para o conjunto de instâncias da empresa B *(continuação da página anterior).*

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
CUC28A_133	300	16,4	301	849,1	110	301	1132	114
GTM18AT_134	-	-	187	9,1	34	186	17,9	31
KIT18N_135	26	1,2	26	2,8	42	26	15,3	32
MAC_136	4	0	4	0	1	4	0	0
MARAF1A_137	56	9,4	56	49,3	91	56	338	88
MARAF1R_138	9	0,5	9	0,9	23	9	1,6	21
MOBILTUR_139	231	0,1	231	0,5	15	232	0,6	12
REVAL_140	*	62,8	282	1467,3	338	-	-	-
RIO18G_141	15	0	15	0	4	15	0	1
RIO18N_142	192	5,3	193	674,7	94	193	686,3	95
RIO25E_143	6	0	6	0	1	6	0	0
RIO25G_144	36	0	36	0,1	5	36	0	3
RIO25N_145	144	0,2	145	2,2	32	145	1,2	27
STE1A_146	5	0	6	0,1	5	6	0,1	5
STE1R_147	3	0	3	0,4	2	3	0	2

Instâncias adaptadas da literatura

Os resultados computacionais obtidos com as heurísticas neste conjunto de instâncias foram bastante piores do que nos conjuntos de instâncias reais. A heurística com o modelo de geração de colunas 1 obteve a solução ótima em apenas 5 instâncias, enquanto o modelo de geração de colunas 2 não obteve nenhuma. Além disso, em nenhuma das 5 vezes em que a heurística com o modelo de geração de colunas 1 encontrou a solução ótima, a solução foi obtida mais rapidamente do que o modelo de programação inteira

Tabela 3.20: Resultados para o conjunto de instâncias C.

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
A1	23	0.0	23	1	23	23	0,7	20
A2	12	0.1	13	2,6	43	13	1,4	32
A3	8	0.2	10	4,1	52	9	2,4	47

Tabela 3.20 Resultados para o conjunto de instâncias *C (continua da página anterior).*

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
A4	5	0.9	7	8	65	7	5,4	49
A5	5	0.9	6	7,2	64	6	6,4	54
CHL1	6	54.9	7	32,2	123	9	45,1	116
CHL2	3	0.1	4	0,6	22	4	0,5	23
CHL5	4	0.0	5	0,2	12	5	0,3	15
CHL6	6	4.6	7	36,5	140	8	38	130
CHL7	6	15.0	8	61,2	168	8	61,4	153
CU1	12	0.4	14	6,6	55	15	3,8	58
CU2	15	0.3	16	14,7	107	16	7,8	88
CW1	10	0.3	11	6,7	70	11	4	65
CW2	12	4.2	13	10,5	98	13	9	104
CW3	16	3.0	17	24,4	126	17	12,6	117
Hchl2	6	88.4	8	53,2	153	8	52,8	144
Hchl3s	3	0.7	4	1,7	37	4	1,4	32
Hchl4s	2	0.7	3	1,6	36	3	1,4	31
Hchl6s	5	6.7	6	11,6	79	7	10,6	74
Hchl7s	7	75.2	9	101,3	208	10	177,3	199
Hchl8s	2	0.1	2	0,9	25	4	0,6	10
Hchl9	10	5.3	12	28	123	12	19,1	113
HH	2	0.0	2	0,2	11	2	0,1	12
OF1	4	0.0	4	0,7	22	5	0,3	15
OF2	5	0.0	6	0,3	14	6	0,3	13
STS2	12	1.7	13	15,8	116	13	7,8	84
STS4	5	1.2	6	9,6	81	7	6,7	68
W	24	0.1	24	2,3	38	25	1,7	38
2	2	0.3	5	0,5	19	6	0,4	21
3	23	0.1	24	1	30	23	0,8	24

O conjunto de instâncias D cujos resultados são apresentados na tabela 3.21 revelou-se o grupo mais difícil de resolver. O modelo de programação inteira encontrou a solução ótima em apenas 60% das instâncias e a heurística para os modelos de geração de colunas não conseguiu atingir nenhuma solução ótima.

Tabela 3.21: Resultados para o conjunto de instâncias D.

Instância	Modelo proposto		Modelo de geração de colunas 1			Modelo de geração de colunas 2		
	Z	T_{opt}	Z	T_{H1}	N_{col}	Z	T_{H2}	N_{col}
ATP30	9	206.1	13	567,5	118	13	616,4	122
ATP31	15 (1)	7257.8	17	3068,8	202	17	3557	206
ATP32	13	893.7	21	1115,6	185	20	1072,8	175
ATP33	13 (1)	7210.6	15	795,7	204	15	708	188
ATP34	6	303.6	9	159,1	95	9	187,5	85
ATP35	8	415.2	10	280,2	122	10	380,8	127
ATP36	8	220.2	10	174,3	103	10	148,3	85
ATP37	12	2496.2	16	782,4	155	15	988,9	156
ATP38	11	149.6	15	712,2	134	15	732,6	131
ATP39	12 (1)	7206.8	13	328,4	145	13	358,3	143
ATP40	16 (1)	7232.1	21	3969,3	206	20	4177,1	217
ATP41	12	29.2	14	546,2	151	14	581,3	155
ATP42	16 (1)	7212.7	20	2838,6	264	20	4889,9	231
ATP43	14 (1)	7224.8	17	1628,1	202	17	1457,5	177
ATP44	9	3583.9	13	376,5	139	13	457,3	133
ATP45	8	1724.8	11	198,9	144	10	273,3	152
ATP46	12 (1)	7216.1	14	909	181	14	1004,1	180
ATP47	13	778.2	15	932,6	211	14	1005,4	206
ATP48	9 (1)	7218.7	11	918,7	131	11	993,6	137
ATP49	5	4271.2	8	186	90	8	236,4	98

3.5 Conclusões

Neste capítulo, foi proposto um modelo de programação inteira para resolver os problemas de corte bidimensional de dois e de três-estágios. O modelo é uma extensão do modelo "*one cut*" proposto por Dyckhoff (1981) para o problema de corte a uma dimensão. Um corte corresponde a cortar um item de uma placa, que pode ser uma placa inicial ou uma placa residual, produzindo mais placas residuais com diferentes tamanhos de acordo com o estágio em que é executado.

Embora seja de tamanho pseudo-polinomial, o modelo é otimizado por um *solver* geral de programação inteira (Cplex11.0 2007) que resolveu um grande número de instâncias reais em tempos computacionais aceitáveis. Os tempos computacionais comprovam que o modelo é sensível ao número de tipos de itens

da instância e da relação entre o tamanho da placa e os tamanhos dos itens (instâncias em que as placas acondicionam poucos itens são mais fáceis para qualquer abordagem de solução), mas não é sensível ao número de itens.

Para as 672 combinações de instâncias reais e problemas testados (168 instâncias para cada problema), apenas em 29 casos, correspondendo a instâncias com um número grande de tipos de itens, não foi possível encontrar a solução ótima em duas horas. Em 21 destes casos, a solução ótima ou é a solução incumbente ou usa apenas uma placa a menos que a solução incumbente. Para as instâncias reais, o comportamento dos modelos de afetação da literatura (que podem também ser resolvidos diretamente por um *solver* de programação inteira) foram claramente inferiores ao modelo proposto, uma vez que estes modelos são sensíveis ao número de itens da instância e algumas instâncias têm milhares de itens. Para as instâncias adaptadas da literatura o comportamento dos dois modelos foi similar.

O modelo proposto superou claramente os modelos de afetação existentes na literatura, assim como os dois modelos de geração de colunas testados.

A flexibilidade do modelo proposto permite considerar outras características que podem ter um papel relevante em problemas práticos. Em particular, a existência de placas de vários tamanhos, a incorporação de valores para as placas remanescentes, a inclusão dos comprimentos dos cortes e a rotação dos itens foram também abordados.

Capítulo 4

Problema integrado de corte bidimensional e dimensionamento de lotes [1]

O problema de corte bidimensional consiste na minimização do número de placas utilizadas para o corte de um conjunto de itens retangulares. Tipicamente, na indústria, uma instância deste problema é considerada no início de cada período de planeamento, o que pode resultar em soluções de fraca qualidade, isto é, com desperdício excessivo, quando considerado um conjunto de períodos de planeamento. Com o objetivo de abordar esta característica, é proposto neste capítulo um problema integrado, em que o problema de corte bidimensional é estendido da resolução em apenas um período de planeamento para a resolução num conjunto de períodos de planeamento. A principal diferença entre a abordagem apresentada neste capítulo e as utilizadas na literatura é permitir o armazenamento de objetos residuais (*sobras*) para serem cortadas em períodos subsequentes do horizonte de planeamento, o que poderá ajudar ainda mais na minimização dos desperdícios. Deste modo, são propostos dois modelos de programação inteira que otimizam o problema combinado de corte a duas dimensões e o dimensionamento de lotes, minimizando os desperdícios e os custos de armazenamento. São apresentados resultados computacionais para os modelos propostos.

[1] Uma versão preliminar e simplificada do trabalho descrito neste capítulo foi publicado em Silva et al. (2011*b*). O trabalho descrito neste capítulo foi submetido a uma revista (Silva et al. 2011*a*).

4.1 Introdução

Os problemas de corte bidimensionais têm sido amplamente estudados nos últimos cinquenta anos, devido à sua vasta aplicabilidade prática. No entanto, na maioria das industrias estes problemas não são estudados como fazendo parte de um processo integrado (Erjavec et al. 2009). O planeamento do processo de corte é repetido no início de cada período de tempo, apenas para atender às procuras dos clientes no período de tempo considerado.

A procura de uma solução ótima para o problema de *2CSP* em cada período do horizonte de planeamento é diferente de uma avaliação completa de todo o horizonte de planeamento. A antecipação da produção de itens e o seu armazenamento em inventário para utilização em períodos subsequentes do horizonte de planeamento corresponde à mudança de uma política local para uma estratégia ótima global. Deste modo, a eficiência da produção é melhorada e os custos são reduzidos, às custas de se ter um problema mais difícil de resolver.

Neste capítulo será estudado o problema integrado de corte a duas dimensões e o problema de dimensionamento de lotes (*2CS-LSP*). Dada uma procura de itens retangulares durante um dado horizonte de planeamento, o problema é determinar a quantidade de itens que deve ser cortada das placas retangulares em cada período do horizonte de planeamento, de modo a que os custos de produção e de armazenamento e os desperdícios das placas sejam minimizados.

Ambos os problemas *2CSP* e dimensionamento de lotes (*LSP*) são NP-difíceis, e sendo assim a sua integração é também um problema NP-difícil.

No problema de *2CSP* estudado existe um conjunto de placas retangulares do mesmo tamanho, isto é, com a mesma largura e altura, de onde se pretende cortar um conjunto de itens retangulares mais pequenos. As placas grandes estão disponíveis em número (virtualmente) infinito e o conjunto de itens a cortar está agrupado por tipo, de acordo com o tamanho (largura e altura). O objetivo é

minimizar o número de placas grandes utilizadas ou o desperdício. No problema de *LSP* multi-item considerado, pretende-se determinar as quantidades a produzir de cada item num dado número de períodos de tempo, de modo a satisfazer as procuras de cada item em cada período, minimizando os custos de armazenamento e de produção.

Resolver estes dois problemas separadamente pode levar a soluções não-ótimas para o problema integrado. Integrar os problemas *2CSP* e *LSP* pode ser visto como um balanceamento entre antecipar a produção de alguns itens, fornecendo uma melhor combinação dos itens nas placas, promovendo assim a minimização dos desperdícios, e não antecipar a produção dos itens, fomentando a minimização dos custos de armazenamento.

Tal como será detalhado abaixo, na secção revisão da literatura, alguns autores já abordaram a antecipação da produção e de inventário de placas completas e de itens. No entanto, podem existir situações nas quais os objetos residuais (*sobras*) de um dado período sejam considerados desperdícios, porque o seu armazenamento não é considerado. Neste capítulo, são apresentados novos modelos que permitem armazenar itens e objetos residuais (e também placas completas), podendo as sobras ser utilizadas em períodos posteriores num padrão de corte do tipo dois-estágios não-exato. Claramente, este modelo pode fornecer soluções com uma melhor utilização da matéria-prima.

Dois métodos que permitem obter soluções ótimas para o problema integrado de *2CS-LSP* são propostos. O primeiro modelo de programação inteira resulta de uma extensão do modelo de programação inteira para o problema de corte bidimensional proposto no capítulo anterior. O segundo modelo de programação inteira resulta também de uma extensão, mas neste caso, diretamente do modelo proposto por Dyckhoff (1981) para o problema unidimensional.

Os modelos de programação inteira propostos permitem resolver otimamente

o problema integrado de *2CS-LSP*, modelando a antecipação da produção dos itens e também o armazenamento de objetos que sobraram dos cortes, no final de cada período do horizonte de planeamento. Os modelos propostos não têm por base a geração de padrões de corte.

Este capítulo inicia-se com uma revisão da literatura sobre problemas combinados na secção 4.2. Os modelos de programação inteira para o problema combinado são propostos na secção 4.3 assim como um pequeno exemplo. As duas últimas secções são dedicadas à apresentação de resultados computacionais e às conclusões.

4.2 Revisão da literatura

Não existem na literatura muitos artigos com o estudo da integração dos problemas de *2CS-LSP*, e a maioria dos trabalhos está muito relacionada com a aplicação industrial em que surge o problema.

Possivelmente, o primeiro trabalho que aborda o problema combinado foi o de Farley (1988). O problema surge na indústria do vestuário, onde um modelo de programação inteira e um modelo de programação quadrática foram propostos. O principal objetivo é minimizar os custos globais de produção, tendo em conta os custos de corte, de costura, de armazenagem e também de produção em excesso e produção em falta. No entanto, o problema integrado de corte e dimensionamento de lotes não foi explicitamente resolvido, sendo em vez disso o problema formulado como um problema de planeamento da produção ao longo de um único período.

O processo de produção na indústria do cobre foi também analisado de forma integrada por Hendry et al. (1996). Um procedimento em duas fases foi proposto: na primeira fase uma solução agregada é determinada para o problema de corte a

uma dimensão, na segunda fase é decidido o escalonamento da produção para cada dia do período de planeamento. Procedimentos heurísticos foram utilizados para resolver o primeiro estágio e um modelo de programação inteira foi considerado no segundo estágio.

Um problema de *2CSP* com figuras irregulares e um problema de dimensionamento de lotes com procura estocástica, com origem numa empresa de máquinas basculantes (*dumpers*), foram resolvidos de forma integrada por Nonas & Thorstenson (2000). A minimização dos desperdícios, dos custos de armazenagem e de preparação foram o principal objetivo. O problema foi formulado de modo não-linear, com uma função objetivo côncava com restrições lineares. Foram utilizadas heurísticas para resolver o problema e também um procedimento de geração de colunas. Posteriormente em Nonas & Thorstenson (2008) a abordagem proposta foi integrada com a heurística sequencial apresentada por Haessler (1971).

O problema de corte a uma dimensão surge integrado na alocação de encomendas em máquinas paralelas em Menon & Schrage (2002). Os autores tiram vantagem da estrutura angular do problema e com o objetivo de reduzir o efeito de simetria desenvolveram um procedimento baseado na construção de limites apertados nos subproblemas. Em Arbib & Marinelli (2005) o problema de corte a uma dimensão foi integrado com o problema de planeamento do inventário. Os autores propõem um modelo de programação inteira e uma metodologia para integrar um sistema hierárquico de planeamento da produção. O modelo é uma extensão de um otimizador de dia-a-dia, permitindo a inclusão de custos de transporte e de armazenagem, que pode ser utilizado para gerir "encomendas prioritárias" (quando chega uma encomenda depois do plano de produção estar completo).

Na indústria do papel o problema integrado de corte e dimensionamento de lotes foi abordado por Poltroniere et al. (2008). Os autores propõem um modelo de programação inteira e duas heurísticas de decomposição. A primeira heurística

que propõem baseia-se na relaxação lagrangeana, o problema de dimensionamento de lotes é resolvido primeiro e depois é resolvido o problema de corte. As variáveis duais das restrições de ligação são utilizadas para redefinir os custos para o de dimensionamento de lotes. A outra heurística começa por resolver o problema de corte, determinando o número total de jumbos necessários por tipo de papel e período, o planeamento da produção é obtido resolvendo o problema de dimensionamento de lotes.

O problema *2CS-LSP* foi também tratado por Gramani & França (2006), onde o método proposto consiste em determinar as quantidades a produzir de itens em cada período do horizonte de planeamento, de tal modo que o número de placas utilizadas, os custos de armazenamento e de preparação sejam minimizados. Um modelo de programação inteira com limite de tempo na capacidade da serra foi proposto, mas não foi considerado o armazenamento de objetos residuais para serem utilizados em períodos subsequentes do horizonte de planeamento. Foi proposto um modelo de programação inteira, mas no entanto foram realizados apenas testes computacionais com uma heurística utilizando uma representação numa rede de caminho mais curto. Cada arco da rede está associado com a capacidade do *2CSP*, resolvido através do procedimento de Gilmore & Gomory (1965) de geração de colunas. Depois de resolvidos todos os arcos, é resolvido um problema de caminho mínimo. Nesta abordagem, em vez da antecipação individual dos itens, é antecipada a produção de todos os itens de um dado cliente. O procedimento heurístico foi comparado com o método utilizado tradicionalmente na indústria, que consiste em resolver primeiro o problema de dimensionamento de lotes e depois, num segundo estágio, o problema de corte. Os autores concluíram que resolver a versão combinada do problema permite benefícios nos lucros acima dos 28%.

Uma variante diferente foi considerada em Gramani et al. (2009). Neste tra-

balho a restrição de capacidade de tempo e os custos de preparação de Gramani & França (2006) são mantidos, não se considerando o armazenamento de objetos residuais e a procura em produtos finais, sendo o armazenamento não em itens individuais mas em produtos finais. É proposto um modelo de programação inteira resolvido através de um procedimento heurístico baseado em relaxação lagrangeana.

Outra variante foi proposta em Gramani et al. (2010); neste caso os custos de preparação não foram considerados assim como as frequências inteiras dos padrões de corte, contudo é considerado o armazenamento não apenas de produtos finais, mas também de itens individuais. Um modelo de programação linear baseado em Gramani et al. (2009) foi utilizado e resolvido através de geração de colunas.

Em Poldi (2007) um problema semelhante foi estudado como um problema de corte multi-período, que considera o balanceamento entre antecipar ou não a produção de alguns itens no horizonte de planeamento. As placas iniciais, que estão disponíveis em diferentes tamanhos e em quantidades limitadas, que não foram utilizadas num dado período de tempo são guardadas em inventário e poderão ser utilizadas nos períodos de tempo seguintes. Contudo, o armazenamento de objetos residuais para serem cortados em períodos subsequentes não foi considerado. Propõem um modelo de programação inteira em que as soluções da relaxação linear são encontradas através de geração de colunas e as soluções inteiras são obtidas heuristicamente.

4.3 Modelos de programação inteira

Nesta secção são apresentados dois modelos de programação inteira para o problema *2CS-LSP*. Ambos os modelos têm como ponto de partida o modelo para o problema de corte unidimensional de Dyckhoff (1981). O Modelo I resulta de

uma extensão prévia do modelo para o problema de corte bidimensional apresentado no capítulo anterior, enquanto o Modelo II é uma nova extensão para a formulação do problema de corte unidimensional.

O modelo de Dyckhoff (1981) é bastante flexível, permitindo considerar extensões de modo a ter em conta características que poderão ter um papel muito relevante em problemas práticos. A extensão para o problema bidimensional foi proposta no capítulo anterior, e os resultados computacionais obtidos para instâncias da literatura e instâncias reais foram muito promissores.

4.3.1 Modelo I

O modelo proposto é uma extensão do modelo de programação inteira apresentado no capítulo anterior para o problema *2CSP*.

O conceito central do modelo foi denominado por *corte* e consiste num conjunto de operações elementares (um ou dois cortes guilhotinados), que obtêm um item de uma placa. Tipicamente, de um corte resulta um item e duas placas que poderão posteriormente ser cortadas (*placas residuais*).

O modelo baseia-se na enumeração de todos os cortes e tipos de placas residuais possíveis, utilizando um conjunto de regras simples para calcular os cortes, a largura, a altura e o estágio das placas residuais resultantes. A ideia principal do modelo de programação inteira é associar uma variável de decisão com cada corte, isto é, com o número de vezes que um corte é executado.

O algoritmo para a geração de todos os cortes e placas residuais, assim como limites superiores para o número de tipos de placas residuais e cortes foram apresentados no capítulo 3. No problema integrado de corte e dimensionamento de lotes considerado neste capítulo, os padrões de corte são do tipo dois-estágios não-exato.

Na figura 4.1 são apresentados dois tipos de cortes: no caso A, os cortes

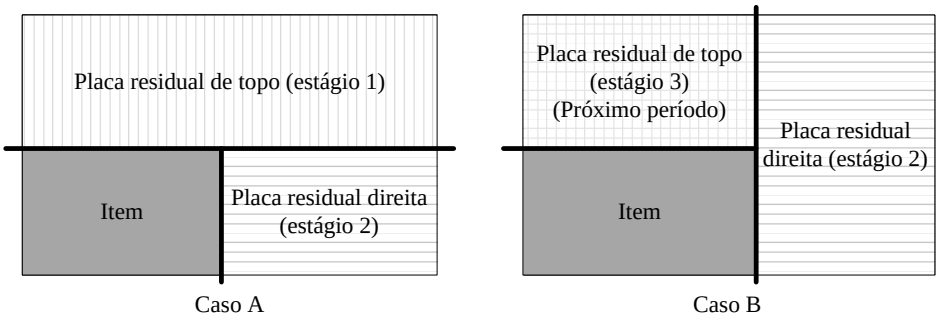


Figura 4.1: Exemplos de corte.

(linhas a negrito) podem ser efetuados em placas iniciais ou em placas residuais no estágio um, enquanto os tipos de cortes apresentados no caso B podem apenas ser executados em placas residuais no estágio dois. Tendo em conta que os padrões considerados para *2CS-LSP* são do tipo dois-estágios não-exato, a placa residual no estágio três no caso B não será considerada no período de planeamento atual para dela serem cortados itens, sendo em vez disso armazenada em inventário como um objeto residual, que apenas pode ser cortado no período de tempo seguinte, sendo então considerada como uma placa inicial.

Nos padrões de corte do tipo dois-estágios não-exato as placas residuais podem ser consideradas sobras se não forem cortadas no período atual do planeamento, contudo existe um tipo de placa residual que não pode ser cortada no período de tempo em que foi obtida, a placa residual de topo no estágio três, porque caso contrário o padrão de corte não seria do tipo dois-estágios não-exato. As placas residuais de topo no estágio três são consideradas sobras no período de tempo em que são obtidas, e assim são guardadas em inventário e ficam disponíveis para serem cortadas no período de tempo seguinte, sendo então consideradas como

placas iniciais.

O modelo de programação inteira proposto para o problema *2CS-LSP* baseia-se no modelo de programação inteira para o problema *2CSP*, as principais diferenças residindo na consideração de períodos de tempo, na utilização de variáveis de decisão associadas com o armazenamento de itens e de placas residuais como também nos parâmetros relacionados com os custos.

A partir daqui assume-se que todos os cortes possíveis assim como os tipos de placas residuais são conhecidos para cada período.

No problema *2CS-LSP* as placas retangulares iniciais têm largura W e altura H e o item procurado i a ser cortado no período t, tem largura w_i, altura h_i e procura d_i^t. O número de tipos de itens é novamente definido como n, o número de tipos de placas residuais em cada período é definido como m_t e o número de períodos de tempo por T. Os tipos de placas são indexados por j, $j = 0, ..., m_t$, com $j = 0$ correspondendo à placa inicial e $j = 1, ..., m_t$, correspondendo aos tipos de placas residuais. Os parâmetros e variáveis do modelo são definidos como:

Parâmetros:

d_i^t: procura do item do tipo i no período t;

c_i^t: custo de cortar o item do tipo i no período t;

g_i^t: custo de guardar em inventário o item do tipo i no período t;

h_j^t: custo de guardar em inventário a placa residual do tipo j no período t;

c_0^t: custo da placa inicial no período t;

a_{ijk}^t: igual a 1 se a placa do tipo k resulta de cortar o item do tipo i da placa do tipo j no período de tempo t, e igual a 0 caso contrário;

$K = \{$placa k: estágio de $k = 3$ e $a_{ijk}^{t-1} = 1\}$ a placa do tipo k no estágio 3 que foi obtida de cortar o item do tipo i da placa do tipo j no período de tempo $t-1$;

$L = \{$placa k: estágio de $k \neq 3\}$ uma placa do tipo k que está no estágio 1 ou 2

b_{ij}: igual a 1 se o item do tipo i cabe na placa do tipo j, $j \in K$ ou $j \in L$, e igual a 0 caso contrário;

e_{ik}: igual a 1 se o item do tipo i cabe na placa do tipo k, $k \in K$, e igual a 0 caso contrário;

Variáveis de decisão:

x_{ij}^t: número de vezes que o item do tipo i é cortado da placa do tipo j no período t;

I_i^t: número de itens do tipo i em inventário no final do período t

s_j^t: número de placas do tipo j em inventário no final do período t;

Pretende-se definir a quantidade de itens a cortar em cada período de tempo, decidindo se o corte de alguns itens deve ou não ser antecipado, de modo que os custos do processo de corte e de armazenamento e o custo das placas iniciais utilizadas sejam minimizados, respeitando a procura de cada item em cada período do horizonte de planeamento. Utilizando a notação previamente apresentada, o modelo de programação inteira para o problema *2CS-LSP* é o seguinte:

$$\text{Minimizar} \quad \sum_{t=1}^{T}\sum_{i=1}^{n}(c_0^t + c_i^t)x_{i0}^t + \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=1}^{m_t} c_i^t x_{ij}^t + \sum_{t=1}^{T}\sum_{j=1}^{m_t} h_j^t s_j^t + \sum_{t=1}^{T}\sum_{i=1}^{n} g_i^t I_i^t \tag{4.1}$$

$$\text{Sujeito a} \quad \sum_{j=0}^{m_t} b_{ij}x_{ij}^t + I_i^{t-1} - I_i^t = d_i^t, \qquad i = 1, ..., n, t = 1, ..., T \tag{4.2}$$

$$\sum_{j=0}^{m_t}\sum_{i=1}^{n} a_{ijk}^t x_{ij}^t - \sum_{i=1}^{n} b_{ik} x_{ik}^t + s_k^{t-1} - s_k^t = 0,$$
$$k = 1, .., m_t; t = 1, ..., T \tag{4.3}$$
$$-\sum_{i=1}^{n} e_{ik} x_{ik}^t + s_k^{t-1} \geq 0, \qquad k \in K, t = 1, ..., T \tag{4.4}$$
$$x_{ij}^t \geq 0 \text{ e inteiro}, \qquad i = 1, .., n; j = 0, ..., m_t; t = 1, ..., T \tag{4.5}$$
$$s_j^t \geq 0 \text{ e inteiro}, \qquad j = 1, ..., m_t; t = 1, ..., T \tag{4.6}$$
$$I_i^t \geq 0 \text{ e inteiro}, \qquad i = 1, ..., n; t = 1, ..., T. \tag{4.7}$$

Na função objetivo (4.1) são considerados os custos de produção dos itens, o custo das placas inicias c_0^t que podem variar em cada período do horizonte de planeamento, dependendo da flutuação de preços da matéria prima nos mercados, os custos de armazenamento das placas residuais, e os custos de armazenar os itens cuja produção foi antecipada.

As restrições (4.2) garantem que a procura de cada tipo de item em cada período é satisfeita, balanceando os itens que são cortados no respetivo período, de placas iniciais ou residuais, os itens que estão em inventário de períodos anteriores e os itens que serão armazenados em inventário no final do período de planeamento considerado, garante-se também que os cortes não são feitos em placas no estágio 3 obtidas no período de tempo em análise, através da definição do parâmetro b_{ij}.

O segundo grupo de restrições (4.3) determina que para cada tipo de placa residual em cada período (exceto placas residuais no estágio 3 obtidas no período de tempo atual), o número de cortes que dão origem a esse tipo de placa residual no período considerado mais o número de placas residuais do mesmo tipo guardadas em inventário no período de tempo anterior são iguais ao número de cortes realizados nessas placas residuais mais as placas residuais desse tipo que serão guardadas em inventário no final do período, desta forma garante-se que serão apenas realizados cortes em placas residuais existentes.

Finalmente, o último conjunto de restrições (4.4) garante que é apenas possível cortar placas residuais no estágio 3 que tenham como origem o inventário. Estas restrições garantem que os padrões serão do tipo dois-estágios não-exato.

4.3.2 Modelo II

O segundo modelo de programação inteira proposto está relacionado com uma abordagem do problema de corte bidimensional de dois-estágios não-exato como uma combinação de dois tipos de problemas de corte a uma dimensão.

Como visto no capítulo anterior com os modelos de geração de colunas, padrões de corte do tipo dois-estágios não-exato podem ser obtidos acondicionando os itens horizontalmente, construindo tiras horizontais e depois agrupando-as verticalmente.

No modelo proposto, a construção das tiras horizontais e a sua integração vertical são tratados como um conjunto de problemas a uma dimensão, resolvidos através da formulação de Dyckhoff (1981).

Tal como já foi referido ao longo dos capítulos anteriores, neste modelo cada variável corresponde a uma operação de corte única executada num único objeto. Um objeto de qualquer tamanho é cortado em duas peças e pelo menos uma delas tem o tamanho de um item procurado. Uma operação de corte pode ser realizada em objetos grandes guardados em inventário ou em peças residuais resultantes de cortes anteriores.

A construção das tiras horizontais é feita resolvendo um conjunto de problemas de corte unidimensionais, a altura da tira é fixada pela altura do primeiro item, deste modo a otimização centra-se na largura, ver figura 4.2 (tiras horizontais caso B).

Um segundo conjunto de problemas de corte a uma dimensão são resolvidos para decidir quais serão as tiras selecionadas para produzir os padrões de corte,

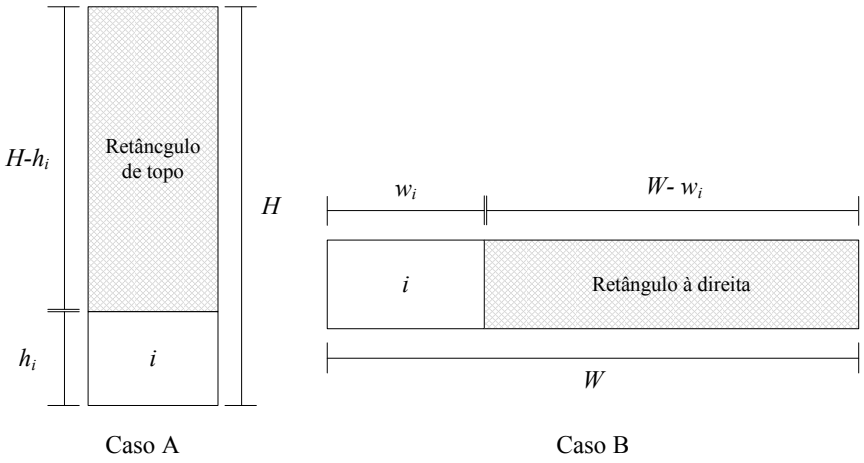


Figura 4.2: Tira vertical (Caso A) e tira horizontal (Caso B).

neste caso a largura é fixada com o valor da largura da placa inicial e apenas a altura é otimizada, ver por exemplo a figura 4.2 tira vertical (caso A).

Todos os cortes e objetos residuais possíveis para cada tipo de problema de corte unidimensional deve ser gerado. Dois algoritmos são apresentados para o cálculo de todos os cortes possíveis e retângulos residuais.

Os outputs do algoritmo 3 são duas listas: a lista R é constituída por todos os retângulos à direita possíveis e a lista C é composta por todos os cortes verticais possíveis nas tiras horizontais.

O algoritmo 3 começa por ordenar os itens por altura e por inicializar as duas listas, depois os cortes são analisados, é verificado se um retângulo à direita é obtido desse corte, com tamanho suficiente para ser cortado novamente. Se tiver tamanho aceitável os cortes possíveis são examinados, para identificar quais os itens que podem ser cortados desse retângulo à direita. Todo o processo é repetido até que todos os cortes sejam analisados. No algoritmo 3 todos os retângulos à direita estão no estágio 2, exceto os que estão no estágio 3 e todos os retângulos

obtidos no algoritmo 4 estão no estágio 1.

O algoritmo 4 para a geração de todos os cortes horizontais possíveis e todos os retângulos de topo é similar ao algoritmo 3, contudo neste caso é apenas considerada a altura.

Algoritmo 3: Algoritmo para a geração de cortes verticais e tipos de retângulos à direita.

início

Ordenar os itens por altura;

Inicializar R com o conjunto vazio;

Inicializar C com n cortes correspondendo a cortar cada i da tira com altura h_i e largura W_l (ver figura 2.2 caso B);

Marcar todos os cortes em C como não analisados;

enquanto *não forem analisados todos os cortes de* C **faça**

Seja c um corte de c não analisado associado com o item do tipo i e um tipo de retângulo à direita;

Marcar c como analisado;

Adicionar o novo tipo de retângulo à direita com altura h_i e largura $W_l - w_i$ a R se este for maior ou igual ao tamanho do item mais pequeno e se ainda não constar na lista;

Adicionar o novo retângulo à direita no estágio 3 com altura $h_l - h_i$ e largura w_i a R se for maior ou igual ao tamanho do item mais pequeno se ainda não constar da lista;

para todo *itens do tipo* $j : h_j \leq h_i$ **faça**

se *item do tipo* l *pode ser cortado do novo retângulo à direita* **então**

Adicionar o corte associado com o novo retângulo à direita e o item do tipo j a C e marcá-lo como não analisado

fim

fim

fim

fim

Algoritmo 4: Algoritmo para a geração de cortes horizontais e retângulos de topo.

início

Inicializar R com o conjunto vazio;

Inicializar C com n cortes correspondendo a cortar cada tira horizontal com altura h_i e largura W da tira vertical com altura H_q (ver figura 2.2 case A);

Marcar todos os cortes em C como não-analisados;

enquanto *não forem analisados todos os cortes de* C **faça**

Seja c um corte não analisado de C associado com a tira horizontal l e um retângulo de topo;

Adicionar o novo retângulo de topo com altura $H_q - h_i$ e largura W a R se $H_q - h_i$ é maior ou igual ao item com menor altura e se ainda não estiver na lista;

para todo *tiras horizontais com largura* W *e altura* h_i **faça**

se *tira horizontal* l *pode ser cortado do novo retângulo de topo* **então**

Adicionar o corte associado com o novo retângulo de topo e a tira horizontal l a C e marcá-lo como não analisado

fim

fim

fim

fim

Formulação

Os parâmetros utilizados anteriormente para os itens e para as placas residuais são adotados. O número de tipos de itens é novamente definido como n, o número de tipos de retângulos à direita é definido como p, o número de tipos de retângulos de topo como q e o número de períodos de tempo como T. Os retângulos à direita são indexados por $j = 0, ..., p$, com $j = 0$ representando o retângulo à direita com largura igual a W; os retângulos de topo são indexados por $r = 0, ..., q$, com $r = 0$ correspondendo ao retângulo de topo com altura H. Os parâmetros e as variáveis de decisão são definidas como:

Parâmetros:

d_i^t: procura do item do tipo i no período de tempo t;

c_i^t: custo de cortar o item do tipo i no período de tempo t;

g_i^t: custo de armazenar o item do tipo i no fim do período de tempo t;

h_j^t: custo de armazenar o retângulo à direita ou de topo do tipo j no fim do período de tempo t;

c_0^t: custo da placa inicial no período de tempo t;

a_{ijk}^t: igual a 1 se o retângulo à direita do tipo k resulta de cortar o item do tipo i do retângulo à direita do tipo j no período de tempo t, 0 caso contrário;

f_{ivr}: igual a 1 se o retângulo de topo do tipo r resulta de cortar a tira horizontal com a mesma altura que o item do tipo i do retângulo de topo do tipo v, 0 caso contrário;

$K = \{$retângulo de topo k: estágio de $k = 3$ e $a_{ijk}^{t-1} = 1\}$ um retângulo de topo do tipo k no estágio 3 que foi obtido através do corte do item do tipo i do retângulo do tipo j no período de tempo $t - 1$;

$L = \{$retângulo k: estágio de $k \neq 3\}$ um retângulo do tipo k que está no estágio 1 ou 2;

b_{ij}: igual a 1 se o item do tipo i cabe no retângulo do tipo j, $j \in K$ ou $j \in L$, e igual a 0 caso contrário;

e_{ik}: igual a 1 se o item do tipo i cabe no retângulo de topo do tipo k, $k \in K$, e igual a 0 caso contrário;

Variáveis de decisão:

z_{ij}^t: número de vezes em que o item do tipo i é cortado do retângulo à direita do tipo j no período de tempo t;

y_{ir}^t: número de vezes que a tira horizontal com a altura que o item do tipo i é cortado do retângulo de topo do tipo r no período de tempo t

I_i^t: número de itens do tipo i em inventário no fim do período de tempo t

s_j^t: número de retângulos à direita do tipo j que são guardados em inventário no fim do período de tempo t;

u_r^t: número de retângulos de topo do tipo r que são guardados em inventário no fim do período de tempo t;

Uma vez que o problema abordado é o *2CS-LSP* o objetivo principal continua a ser decidir a quantidade de itens que devem ser cortados em cada período de tempo, tendo em conta se a sua produção deve ou não ser antecipada, considerando os custos de cortar, os custos de armazenar os itens e os retângulos residuais, e também minimizar o custo das placas utilizadas de modo a que a restrição de procura em cada período seja satisfeita.

Considerando a notação anteriormente introduzida, o segundo modelo de programação inteira para *2CS-LSP* é apresentado a seguir:

$$\text{Minimizar} \quad \sum_{t=1}^{T}\sum_{i=1}^{n}\sum_{j=0}^{p} c_i^t z_i^t + \sum_{t=1}^{T}\sum_{i=1}^{n} c_0^t y_{i0}^t + \sum_{t=1}^{T}\sum_{i=1}^{n} g_i^t I_i^t + \sum_{t=1}^{T}\sum_{j=1}^{p} h_j^t s_j^t + \sum_{t=1}^{T}\sum_{r=1}^{q} h_r^t u_r^t \tag{4.8}$$

$$\text{Sujeito a} \quad \sum_{j=0}^{p} b_{ij} z_{ij}^t + I_i^{t-1} - I_i^t = d_i^t, \qquad i = 1, ..., n; t = 1, ..., T \tag{4.9}$$

$$\sum_{r=0}^{q} y_{ir}^t - z_{i0}^t = 0, \qquad i = 1, ..., n; t = 1, ..., T \tag{4.10}$$

$$\sum_{j=0}^{p}\sum_{i=1}^{n} a_{ijk}^t z_{ij}^t - \sum_{i=1}^{n} b_{ik} z_{ik}^t + s_k^{t-1} - s_k^t = 0, \quad k = 1, ..., p; t = 1, ..T \tag{4.11}$$

$$\sum_{v=0}^{q}\sum_{i=1}^{n} f_{ivr} y_{ir}^t - \sum_{i=1}^{n} b_{iv} y_{iv}^t + u_v^{t-1} - u_v^t = 0, \quad r = 1, ..., q; t = 1, ..T \tag{4.12}$$

$$-\sum_{i=1}^{n} e_{ik} z_{ik}^t + s_k^{t-1} \geq 0, \qquad k \in K; t = 1, ..., T \tag{4.13}$$

$$z_{ij}^t \geq 0 \text{ e inteiro}, \qquad i = 1, .., n; j = 0, ..., p; t = 1, ..., T \tag{4.14}$$

$$y_{ir}^t \geq 0 \text{ e inteiro}, \qquad i = 1, .., n; r = 0, ..., q; t = 1, ..., T \tag{4.15}$$

$$I_i^t \geq 0 \text{ e inteiro}, \qquad i = 1, ..., n; t = 1, ..., T \tag{4.16}$$

$$s_j^t \geq 0 \text{ e inteiro}, \qquad j = 1, ..., p; t = 1, ..., T \tag{4.17}$$

$$u_r^t \geq 0 \text{ e inteiro}, \qquad r = 1, ..., q; t = 1, ..., T \tag{4.18}$$

A função objetivo (4.8) reflete a minimização dos custos de cortar os itens, o custo das placas iniciais, e também os custos de armazenar itens e retângulos residuais.

O primeiro conjunto de restrições (4.9) garante que a procura em cada período é respeitada, balanceando os itens que são cortados das tiras horizontais e os itens que são guardados em inventário.

As restrições (4.10) garantem que serão apenas executados cortes em tiras horizontais existentes, isto é, apenas se executa um corte com altura h_i numa tira vertical se for usada uma tira horizontal com altura h_i.

As restrições (4.11) estão relacionadas com os retângulos à direita, neste caso é garantido que é apenas possível fazer cortes em retângulos à direita existentes (ou retângulos no estágio 3 de períodos de tempo anteriores, $a_{ijk}^{t-1} = 1$) que resultem do corte de outros retângulos à direita ou de inventário, do período de tempo anterior.

De modo similar, as restrições (4.12) estão relacionadas com retângulos de topo: o número de retângulos de topo do tipo v que são cortados, mais os que estão em inventário têm que ser iguais ao número de retângulos de topo existentes, que são gerados de cortes anteriores ou de inventário.

Tal como no modelo I, o último conjunto de restrições (4.13) garante que é apenas permitido cortar placas residuais no estágio 3 que provenham de inventário, garantindo padrões de corte do tipo dois-estágios não-exato.

Exemplo 4.1 De modo a ilustrar os modelos propostos um pequeno exemplo é a seguir apresentado. Considere-se uma instância de *2CS-LSP* com dois períodos de tempo, as placas iniciais com largura e altura igual a 700 e dois tipos de itens, indexados por i, com larguras, alturas e procuras em cada período dadas na tabela 4.1. O custo de cortar cada item (c_i^t) é dado por 0,01% da sua área, os custos de armazenar (g_i^t) cada item são dados por 2% do custo da operação de corte, de forma similar, os custos de armazenar as placas residuais (h_j^t) é dado por 2% do custo das placas residuais, que é 0,01% da área da placa residual. O custo da placa inicial é 50.

O algoritmo 1 apresentado no capítulo 3, para gerar todos os cortes e tipos de placas foi utilizado para calcular tamanhos e estágios da placas residuais possíveis,

Tabela 4.1: Dados do exemplo.

			Procura	
Tipo de item	Largura	Altura	Período de tempo 1	Período de tempo 2
1	200	500	2	2
2	500	200	5	5

Tabela 4.2: Tamanho e estágio dos tipos de placas.

Tipo de placa	Largura	Altura	Estágio
0	700	700	1
1	700	500	1
2	700	300	1
3	500	500	2
4	500	300	3
5	300	500	2
6	700	200	1

estes valores são apresentados na tabela 4.2.

Considerando o *2CSP* e *LSP* de modo independente, uma solução pode ser obtida resolvendo um *2CSP* em cada período do horizonte de planeamento. Sendo assim, será utilizado o modelo de programação inteira apresentado no capítulo 3 para resolver o problema *2CSP* em cada período.

Na tabela 4.3 está representado o modelo de programação inteira proposto no capítulo 3 para o pequeno exemplo para apenas um período de tempo, pode observar-se a cinzento que a solução ótima tem valor 220 e utiliza 3 placas iniciais, uma vez que $x_{20} = 3$. A procura no segundo período de planeamento é igual à procura no período de tempo um, logo a solução ótima é a mesma. Deste modo, a solução obtida para o problema *2CS-LSP* com dois períodos de planeamento

Tabela 4.3: *2CSP* modelo para o exemplo.

		x_{ij}								
item i		2	2	2	1	2	1	1	1	2
placa j		0	1	2	1	3	3	5	0	6
		60	10	10	10	10	10	10	60	10
$z =$	220	3	1	1	1	0	1	0	0	0
j	i									
1					1		1	1	1	$2 = 2$
2		1	1	1		1				1 $5 = 5$
1		1	-1		-1					$1 \geq 0$
2			1	-1						$0 \geq 0$
3					1	-1	-1		1	$0 \geq 0$
5							1	-1		$1 \geq 0$
6									1	-1 $0 \geq 0$

tem valor $220 \times 2 = 440$ e utiliza 6 placas iniciais.

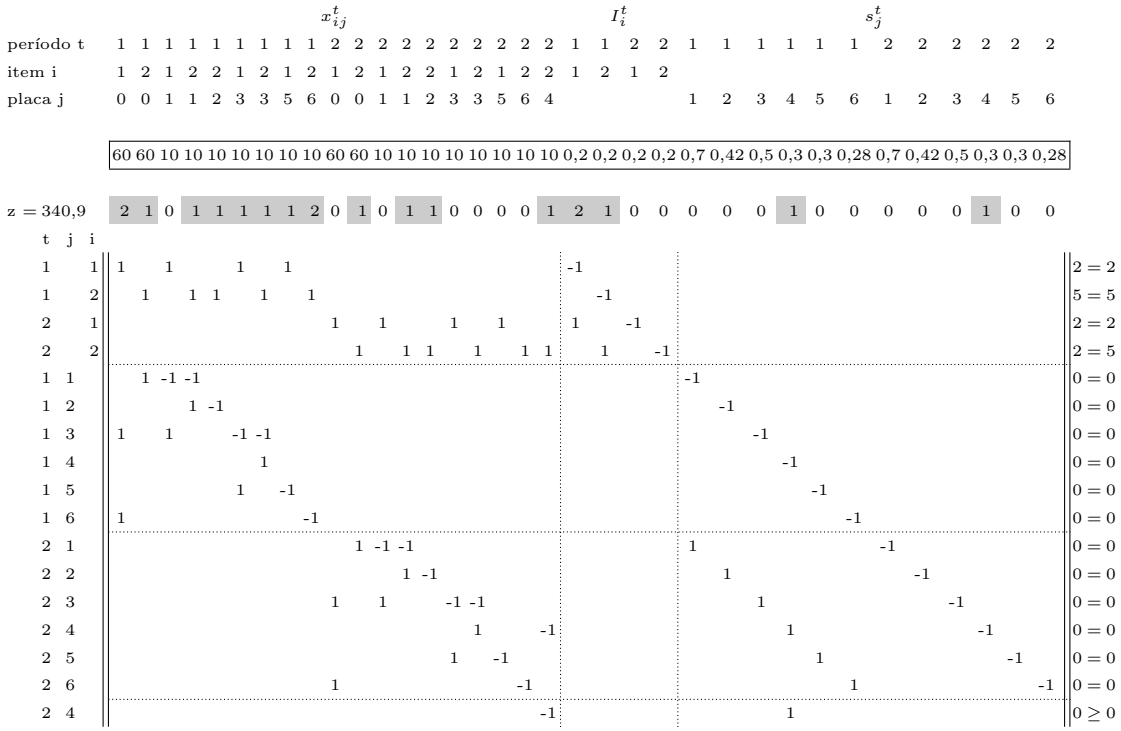
Resolvendo o *2CS-LSP* de uma forma integrada usando os modelos de programação inteira propostos, a solução ótima tem custo 340,9 e utiliza apenas 4 placas iniciais.

O modelo de programação inteira I para o exemplo está apresentado na tabela 4.4, o valor das variáveis de decisão da solução ótima está a cinzento, tal como pode ser observado ($x_{10}^1 = 2$, $x_{20}^1 = 1$ e $x_{20}^2 = 1$), significando que 3 placas iniciais são cortadas no primeiro período de planeamento e 1 placa no segundo período. As variáveis $I_1^1 = 2$ e $I_2^1 = 1$ significam que o corte de dois itens do tipo um e de um item do tipo dois, foi antecipado para o período um, sendo no final deste guardados em inventário.

É importante salientar que a placa residual do tipo 4 está no estágio 3 e $x_{24}^2 = 1$, o que implica que a placa residual do tipo 4 foi obtida no período de tempo um, o que é verdade porque resulta de cortar o item 2 da placa residual 3

($x^1_{23} = 1$). A última restrição garante que a placa residual 4 no período de tempo 2 tem como origem o inventário porque $s^1_4 = 1$.

Tabela 4.4: *2CS-LSP* modelo I para o exemplo.



Para a construção do modelo de programação inteira II para o pequeno exemplo foi necessário calcular todos os retângulos de topo e à direita. Os algoritmos 3 e 4 são utilizados para gerar os retângulos à direita e de topo, respetivamente, podendo os resultados ser encontrados nas tabelas 4.5 e 4.6.

O modelo de programação inteira II para o pequeno exemplo é apresentado na tabela 4.7, e tal como no modelo I o valor ótimo das variáveis de decisão é apresentado em cinzento. Como esperado, utiliza 4 placas iniciais, 3 no período um ($y^1_{20} = 3$) e uma no segundo período de tempo ($y^2_{20} = 1$). As variáveis $I^1_1 = 2$ e $I^1_2 = 1$, significam que o corte de dois itens do tipo um e um item do tipo dois

Tabela 4.5: Tamanho dos retângulos à direita.

Retângulos à direita			
j	w_j	h_j	Estágio
0	700	700	2
1	500	500	2
2	500	300	3
3	300	500	2

Tabela 4.6: Tamanho dos retângulos de topo.

Retângulos de topo			
r	w_r	h_r	Estágio
0	700	700	1
1	700	500	1
2	700	300	1
3	700	200	1

foi antecipado para o período um, sendo guardados em inventário para satisfazer a procura do período seguinte.

O principal propósito deste pequeno exemplo foi expressar a vantagem de resolver os problemas de *2CSP* e *LSP* em conjunto. Tal como era previsível, a estratégia integrada conseguiu diminuir o número de placas iniciais utilizadas e também reduzir os custos.

Tabela 4.7: Modelo II para *2DCS-LSP* para o exemplo.

		z_{ij}^t										y_{ir}^t													I_i^t				s_j^t						u_r^t						
item i		2	1	2	1	1	2	1	2	1	1	2	2	2	1	1	2	2	2	2	1	1	2	2	1	2	1	2													
retângulo à direita j		0	0	1	1	3	0	0	1	1	3													2					1	2	3	1	2	3							
retângulo de topo r												0	1	2	1	0	3	0	1	2	1	0	3												1	2	3	1	2	3	
período t		1	1	1	1	1	2	2	2	2	2	1	1	1	1	1	1	2	2	2	2	2	2	2	1	1	2	2	1	1	1	2	2	2	1	1	1	2	2	2	
		10	10	10	10	10	10	10	10	10	10	50				50		50				50		100,	20,	20,	20,	20,	50,	30,	30,	50,	30,	30,	70,	420,	280,	70,	420,	28	
$z = 340,9$		5	2	1	1	1	3	0	0	0	0	3	1	1	2	0	0	1	1	1	0	0	0	1	2	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
$t\,r\,j$	i																																								
1	1		1		1	1																			-1																2 = 2
1	2	1		1																						-1															5 = 5
2	1							1		1	1														1		-1														2 = 2
2	2						1		1															1		1		-1													5 = 5
1	1		-1												1	1																									0 = 0
1	2	-1										1	1	1			1																								0 = 0
2	1							-1													1	1																			0 = 0
2	2						-1											1	1	1			1																		0 = 0
1 1			1	-1	-1																								-1												0 = 0
1 2				1																										-1											0 = 0
1 3					1	-1																									-1										0 = 0
2 1								1	-1	-1																			1			-1									0 = 0
2 2									1															-1						1			-1								0 = 0
2 3										1	-1																				1			-1							0 = 0
11												1	-1		-1																				-1						0 = 0
12													1	-1																						-1					0 = 0
13																1	-1																				-1				0 = 0
21																		1	-1		-1														1			-1			0 = 0
22																			1	-1																1			-1		0 = 0
23																						1	-1														1			-1	0 = 0
2 2																								-1						1											0>=0

4.4 Resultados computacionais

A construção e otimização dos modelos propostos foram implementados em C++ usando a biblioteca de funções do *solver IBM ILOG Cplex 12.1*. Todos os testes computacionais foram executados num processador de 1.87 GHz com 2GB de memória *RAM*. Em todos os casos foram utilizados os parâmetros por omissão do *Cplex*.

Na tabela 4.8 são apresentadas as principais características das instâncias

reais provenientes de duas empresas que se dedicam à produção de mobiliário: o tamanho das placas iniciais (coluna W e coluna H), o número de tipos de itens (coluna n), a procura média de cada tipo de item (coluna d'), e a largura e altura média de cada tipo de item (colunas w' e h').

Tabela 4.8: Características das Instâncias de *2CS-LSP*.

Instância	W	H	n	d'	w'	h'
10W30_6	4250	1860	3	14	1883,3	1233,3
146O18_17	3760	1860	6	1095	882,7	412,2
15B30_20	4250	2200	5	21,2	1850,8	620
15J30_22	4250	2120	2	28,5	2434,5	484
15X18_25	4250	1860	6	180	1802	462,3
15X22_26	4250	1860	7	428,6	1174,6	450
174N25_31	4250	1860	2	82,5	907	505
26K19_58	2820	1870	2	182,5	723,5	721
26R16_59	2820	1870	2	157,5	730	630
28C38_63	4250	1860	2	29	951	702
28R18_65	4250	1860	2	78	1364	457
30R1_70	4200	1860	1	985	540	900
37A18_73	4250	2070	1	2390	760	490
40C2_82	4250	1860	2	74	631	467,5
42K18_90	4250	1860	3	69,7	1537	403
69W18_111	3760	1860	10	909	602,9	406,7
69Y18_112	3760	1860	7	1830,4	902,7	430,7
76R25_121	3760	1860	2	630	785	435,5
94H18_123	2700	1860	4	1233,8	1687,5	566,3
ARKEN25F_130	4250	2200	1	35	1211	432
ARKEN25P_131	3760	1860	1	400	1211	432
MAC_136	4250	2120	1	32	808	808
RIO25E_143	4250	1860	5	53	845	845
STE1R_147	4200	2200	2	16,5	802	551,5

Os custos do processo de corte para cada tipo de item (c_i^t) é 0,01% da sua área, os custos de armazenagem de cada tipo de item (g_i^t) é 2% do custo de

corte do respetivo item. Os custos de armazenagem das placas residuais (h_j^t) são calculados de forma análoga aos custos de armazenagem dos itens, uma vez que correspondem a 2% do custo das placas residuais ou retângulos residuais (0,01% da sua área). O custo das placas iniciais (c_0^t) foi definido como 50, mas valores diferentes poderiam ser considerados em cada período, refletindo a oscilação dos preços das matérias-primas nos mercados. Foram considerados 2, 5, 7 e 10 períodos de planeamento e o inventário inicial foi considerado nulo. Foi também definido um limite de tempo de 7200 segundos.

Os valores utilizados para os parâmetros são similares aos utilizados por Gramani & França (2006). Na tabela 4.9 e na 4.10 são apresentados para os diferentes períodos de planeamento e para os dois modelos, o tempo total necessário para atingir o ótimo, o que inclui o tempo de construção e o tempo de otimização (coluna t), o tempo despendido apenas pelo *solver* (coluna $t2$), o número de restrições (coluna nc) e o número total de variáveis (coluna nv).

Tabela 4.9: Resultados para os períodos de tempo 2 e 5.

	$t=2$								$t=5$							
	Modelo I				Modelo II				Modelo I				Modelo II			
Instância	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv
10W30_6	0	0	23	47	0,4	0,3	28	53	0,4	0,4	62	122	0,1	0,1	73	137
146O18_17	0,1	0,1	480	2144	0,1	0,1	492	2156	0,9	0,9	1200	5360	1,9	1,9	1230	5390
15B30_20	1,4	1,4	122	500	1,6	1,6	132	510	a	a	305	1250	a	a	330	1275
15J30_22	3,3	3,3	12	30	328,6	328,6	16	34	a	a	30	75	0,1	0,1	40	85
15X18_25	0	0	116	422	0	0	128	434	0	0	290	1055	0	0	320	1085
15X22_26	0,5	0,5	1034	5086	0,8	0,8	1048	5100	14,6	14,6	2585	12715	6,2	6,2	2620	12750
174N25_31	0,7	0,7	52	126	0,5	0,5	56	130	a	a	130	315	a	a	140	325
26K19_58	0,5	0,5	24	58	0,5	0,5	28	62	0,1	0,1	60	145	0,2	0,2	70	155
26R16_59	0	0	20	50	0	0	24	54	0	0	50	125	0	0	60	135
28C38_63	0,4	0,4	32	80	0,3	0,3	36	84	0,5	0,5	80	200	0,4	0,3	90	210
28R18_65	3,4	3,4	26	66	3,5	3,5	30	70	a	a	65	165	7200	7200	75	175
30R1_70	0,1	0,1	16	32	0,1	0,1	18	34	0	0	40	80	0,1	0,1	45	85

Tabela 4.9 Resultados para os períodos de tempo 2 e 5 *(continua da página anterior).*

	$t = 2$								$t = 5$							
	Modelo I				Modelo II				Modelo I				Modelo II			
Instância	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv
37A18_73	0	0	16	32	0	0	18	34	0	0	40	80	0	0	45	85
40C2_82	0,6	0,6	70	182	0,5	0,5	74	186	21,6	21,6	175	455	24,8	24,8	185	465
42K18_90	0,1	0,1	66	182	0,1	0,1	72	188	2,4	2,4	165	455	0,5	0,5	180	470
69W18_111	49,1	49,1	1262	8944	2,1	2,0	1282	8964	a	a	3155	22360	a	a	3205	22410
69Y18_112	1,1	1,1	446	2256	0,1	0,1	460	2270	22,4	22,4	1115	5640	14,8	14,8	1150	5675
76R25_121	0	0	52	132	0	0	56	136	0	0	130	330	0	0	140	340
94H18_123	0	0	46	180	0	0	54	188	0	0	115	450	0	0	135	470
ARKEN25F_130	0	0	14	28	0,1	0,1	16	30	0,3	0,3	35	70	0,3	0,3	40	75
ARKEN25P_131	0,1	0,1	12	24	0,2	0,2	14	26	0,1	0,1	30	60	0,1	0,1	35	65
MAC_136	0,2	0,2	12	24	0,2	0,2	14	26	0,1	0,1	30	60	0,1	0,1	35	65
RIO25E_143	0,2	0,2	12	24	0,1	0,1	14	26	0,3	0,2	30	60	0,3	0,3	35	65
STE1R_147	1,2	1,0	44	118	0,6	0,6	48	122	13,7	13,6	110	295	16,3	16,3	120	305
Total	63,3				340,4				77,5				7266,2			

Para um horizonte de planeamento com dois períodos, o modelo I e o modelo II encontraram a solução ótima em todas as instâncias testadas. Os tempos computacionais são muito similares exceto para a instância 15J30_22 e 69W18_11; em todas as outras instâncias os tempos computacionais são inferiores a 4 segundos.

De um modo geral, à medida que o número de períodos do horizonte de planeamento aumenta, os tempos computacionais também aumentam para ambos os modelos. Contudo, para a maioria das instâncias, os tempos computacionais aumentam pouco, à exceção das instâncias 15J30_22, 174N25_31, 69W18_111, 69Y18_112 e STE1R_147.

[1]Cplex 12.1 retornou "falta de memória".

Tabela 4.10: Resultados para os períodos de tempo 7 e 10.

	$t = 7$								$t = 10$							
	Modelo I				Modelo II				Modelo I				Modelo II			
Instância	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv
10W30_6	0,6	0,6	88	172	0,7	0,7	103	193	9,6	9,6	127	247	0,7	0,6	148	277
146O18_17	29,0	28,9	1680	7504	19,7	19,7	1722	7546	176,1	176,0	2400	10720	a	a	2460	10780
15B30_20	a	a	427	1750	a	a	462	1785	a	a	610	2500	a	a	660	2550
15J30_22	1204,9	1204,9	42	105	43,9	43,9	56	119	a	a	60	150	a	a	80	170
15X18_25	0	0	406	1477	0	0	448	1519	0	0	580	2110	0,1	0,1	640	2170
15X22_26	20,4	20,3	3619	17801	8,8	8,8	3668	17850	36,2	36,1	5170	25.430	301,1	301,1	5240	25500
174N25_31	1576,5	1576,5	182	441	a	a	196	455	a	a	260	630	a	a	280	650
26K19_58	5,3	5,3	84	203	5,1	5,1	98	217	0,9	0,8	120	290	0,6	0,6	140	310
26R16_59	0	0	70	175	0	0	84	189	0	0	100	250	0	0	120	270
28C38_63	1,8	1,8	112	280	1,9	1,9	126	294	12,0	12,0	160	400	14,9	14,9	180	420
28R18_65	a	a	91	231	a	a	105	245	a	a a	130	330	a	a	150	350
30R1_70	0,2	0,2	56	112	0,2	0,2	63	119	0,6	0,5	80	160	0,3	0,3	90	170
37A18_73	0	0	56	112	0	0	63	119	0	0	80	160	0	0	90	170
40C2_82	a	a	245	637	a	a	259	651	a	a	350	910	a	a	370	930
42K18_90	1,8	1,7	231	637	0,6	0,6	252	658	409,5	409,5	330	910	41,1	41,1	360	940
69W18_111	a	a	4417	31304	7200,6	7200,5	4487	31374	a	a	6310	44720	7200,6	7200,5	6410	44820
69Y18_112	515,8	515,8	1561	7896	1436,5	1436,5	1610	7945	a	a	2230	11280	3612,2	3612,2	2300	11350
76R25_121	0	0	182	462	0	0	196	476	0,1	0	260	660	0	0	280	680
94H18_123	0	0	161	630	0	0	189	658	0,1	0	230	900	0,1	0,1	270	940
ARKEN25F_130	0,8	0,8	49	98	3,3	3,3	56	105	1,5	1,5	70	140	6,7	6,7	80	150
ARKEN25P_131	0,9	0,8	42	84	0,8	0,8	49	91	0,4	0,3	60	120	0,4	0,4	70	130
MAC_136	0,5	0,5	42	84	0,6	0,6	49	91	0,3	0,2	60	120	0,3	0,2	70	130

Tabela 4.10 Resultados para os períodos de tempo 7 e 10 *(continua da página anterior).*

	$t = 7$								$t = 10$							
	Modelo I				Modelo II				Modelo I				Modelo II			
Instância	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv	t	$t2$	nc	nv
RIO25E_143	2,7	2,7	42	84	2,4	2,4	49	91	0,4	0,4	60	120	0,5	0,4	70	130
STE1R_147	69,4	69,3	154	413	70,2	70,2	168	427	a	a	220	590	a	a	240	610
Total	3430,7				9866,3				647,7				11179,4			

[1]Cplex 12.1 retornou ”falta de memória”.

O tempo de construção é negligenciável, uma vez que $t - t2$ é muito próximo de zero em quase todas os casos para todos os períodos de planeamento considerados. Para os 192 casos testados foi possível provar a otimalidade em 82% dos testes. O modelo I e o modelo II atingiram o ótimo em 79 dos 96 casos testados. Considerando apenas as soluções ótimas obtidas, o modelo I e o modelo II, atingiram em 98,7% e em 97,5% dos casos o ótimo em menos de 4 segundos, respetivamente.

O modelo I e o modelo II não encontraram a solução ótima em 17 dos 196 casos testados: no modelo II, em 3 das 17 instâncias que não se encontrou o ótimo, tal não aconteceu porque o tempo limite foi alcançado. Nos outros casos a solução ótima não foi encontrada porque o Cplex 12.1 devolveu "falta de memória", no entanto uma solução admissível foi encontrada antes de o *solver* parar.

Os tempos computacionais para a execução das heurísticas não são apresentados porque são negligenciáveis, quase sempre inferiores a um segundo.

Foram também desenvolvidas duas heurísticas. Estas baseiam-se em duas abordagens usadas na prática industrial para resolver o problema. A primeira heurística foi denominada como $H1$ e consiste na antecipação da procura de cada item em cada período para o primeiro período de tempo. Esta abordagem beneficia a minimização do custo das placas utilizadas, contudo os custos de armazenamento aumentam. A segunda heurística utilizada foi a $H2$; nesta abordagem o $2CSP$ é resolvido em cada período do horizonte de planeamento, procurando este procedimento a minimização dos custos de armazenagem.

Na tabela 4.11 as soluções heurísticas e as soluções ótimas são comparadas para horizontes de planeamento com dois e cinco períodos, e na tabela 3.12 esta comparação é efetuada para $t = 7$ e $t = 10$. Nestas tabelas são apresentados o valor da solução (z), o número total de placas usadas (N_{pl}) para $H1$, $H2$ e a solução ótima (Opt). As soluções com (*) são soluções admissíveis dadas pelo

Cplex quando este *solver* para por "falta de memória".

Tal como era expectável a heurística $H1$ utiliza menos placas iniciais do que a heurística $H2$, contudo o valor das soluções de $H1$ são maiores que os valores das soluções obtidas com a heurística $H2$, devido aos custos de armazenagem. Comparando a soma de todas as soluções de $H1$ e a soma de todas as soluções ótimas, em cada período, é obtida uma redução de 1%, 4%, 10% e 8% dos custos nos horizontes de planeamento com 2, 5, 7 e 10 períodos de planeamento, respetivamente.

Os valores das soluções de $H2$ são próximos ou iguais ao ótimo, mas no entanto em cada período de tempo considerado (2, 5, 7 e 10) são utilizadas mais 15, 50, 67 e 102 placas iniciais do que as soluções ótimas, o que significa que com as soluções ótimas foi possível reduzir em 234 o número de placas usadas pela heurística $H2$.

Tabela 4.11: Soluções para os períodos de tempo 2 e 5.

Instância	$t=2$						$t=5$					
	$H1$		$H2$		Opt		$H1$		$H2$		Opt	
	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z
10W30_6	20	15200,60	20	150600	19	15019,84	49	39006,00	50	376500	46	37529,70
146O18_17	930	630848,30	930	625062,68	930	625062,68	2325	1620512,96	2325	1562656,70	2325	1562656,70
15B30_20	31	24325,40	32	24149,90	31	24105,33	78	62529,75	80	60374,76	78*	602585,60*
15J30_22	23	14967,06	24	14880,26	23	14853,06	56	38368,68	60	37200,65	56	37038,09
15X18_25	190	144222,28	190	142888,39	190	142888,39	475	370559,82	475	357220,98	475	357220,98
15X22_26	457	326273,67	458	323319,48	457	323319,48	1142	838190,65	1145	808298,70	1142	808298,70
174N25_31	18	14109,79	18	13979,00	18	13983,44	45	36255,40	45	34947,50	45*	34956,11*
26K19_58	70	32145,39	70	31861,77	70	31861,77	173	82390,61	175	79654,43	173	79578,42
26R16_59	70	32221,52	70	31937,15	70	31937,15	175	82686,59	175	79842,88	175	79842,88
28C38_63	11	7684,10	12	7663,46	11	7624,52	27	19715,00	30	19158,66	27	19032,97
28R18_65	21	16575,00	22	16471,29	21	16444,02	51	42515,34	55	41178,22	51*	41015,43*
30R1_70	141	103749,42	142	102842,00	141	102805,39	352	266529,20	355	257105,00	352	256992,98
37A18_73	239	191737,27	240	190007,20	239	189964,65	598	492718,72	600	475018,00	598	474945,30
40C2_82	11	8035,84	12	8011,72	11	7965,94	27	20620,47	30	20029,30	27	19896,19
42K18_90	40	31165,22	40	30876,45	40	30876,45	100	80078,78	100	77191,14	100	77191,14
69W18_111	705	499600,86	706	495053,32	705	495008,94	1763	1283508,64	1765	1237633,31	1764*	12376500*
69Y18_112	1681	1139956,92	1682	1129552,40	1682	1129552,40	4203	2928326,24	4205	2823881,00	4203	2823815,78
76R25_121	120	83310,96	120	82545,50	120	82545,50	300	214018,31	300	206363,76	300	206363,76
94H18_123	2190	1043624,78	2190	1034376,02	2190	1034376,02	5475	2678427,65	5475	25859405	5475	25859405
ARKEN25F_130	5	3948,68	6	3962,06	5	3929,16	12	10121,37	15	9905,16	12	9787,96
ARKEN25P_131	67	45620,68	68	45252,16	67	45214,94	167	117165,62	170	113130,40	167	113005,73
MAC_136	7	4570,11	8	4578,33	7	4547,29	16	11663,66	20	11445,82	16	11271,94

Tabela 4.11 Soluções para os períodos de tempo 2 e 5 *(continua da página anterior).*

Instância	$t=2$						$t=5$					
	$H1$		$H2$		Opt		$H1$		$H2$		Opt	
	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z
RIO25E_143	11	8194,35	12	8168,67	11	8134,42	27	21028,53	30	20421,66	27	20307,45
STE1R_147	5	3104,40	6	3126,14	5	3088,77	11	7897,97	15	7815,35	11	7635,50

Tabela 4.12: Soluções para os períodos de tempo 7 e 10.

Instância	$t=7$						$t=10$					
	$H1$		$H2$		Opt		$H1$		$H2$		Opt	
	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z
10W30_6	68	55562,60	70	527100	64	52534,95	97	81477,00	100	753000	91	75040,80
146O18_17	3255	2309217,54	3255	2187719,37	3255	2187719,37	4650	3385666,60	4650	3125313,39	4650	3125313,39
15B30_20	109	89110,15	112	84524,67	109*	843090,93*	155	130646,98	160	120749,52	155*	120526,6*
15J30_22	78	54653,76	84	52080,91	78	51821,55	112	80157,42	120	74401,30	112*	74064,64*
15X18_25	665	528120,93	665	500109,37	665	500109,37	950	774466,74	950	714441,96	950	714441,96
15X22_26	1599	1194506,27	1603	1131618,18	1599	1131618,18	2284	1751486,17	2290	1616597,40	2284	1616597,40
174N25_31	62	51623,09	63	48926,50	62	48915,83	89	75730,55	90	69895,00	89*	69887,46*
26K19_58	243	117372,17	245	111516,20	243	111430,18	346	171871,66	350	159308,86	346	159157,94
26R16_59	245	117751,83	245	1117803	245	1117803	350	172482,47	350	159685,75	350	159685,75
28C38_63	38	28105,45	42	26822,12	38	26650,86	54	41195,87	60	38317,31	54	38059,30

Tabela 4.12 Soluções para os períodos de tempo 7 e 10 *(continua da página anterior).*

Instância	$t = 7$						$t = 10$					
	$H1$		$H2$		Opt		$H1$		$H2$		Opt	
	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z	N_{pl}	z
28R18_65	72	60627,47	77	57649,50	72*	57449,57*	102	88873,51	110	82356,43	102*	82028,36*
30R1_70	493	379852,82	497	359947,00	493	359802,94	704	556993,90	710	5142100	704	513984,92
37A18_73	837	702256,71	840	665025,20	837	664922,15	1195	1029889,24	1200	950036,00	1195	949823,00
40C2_82	38	29397,48	42	28041,02	38*	27862,13*	54	43093,87	60	40058,60	54*	39792,38*
42K18_90	140	114131,65	140	108067,59	140	108067,59	200	167376,68	200	154382,27	200	154382,27
69W18_111	2468	1829084,83	2471	1732686,63	2470*	173279*	3525	2681905,62	3530	2475266,62	–	–
69Y18_112	5884	4172828,40	5887	3953433,40	5886	3953433,40	8405	6117965,57	8410	5647762,00	8407	5647762,00
76R25_121	420	304983,82	420	288909,26	420	288909,26	600	447173,00	600	412727,52	600	412727,52
94H18_123	7665	38145404	7665	3620316,07	7665	3620316,07	10950	5588074,31	10950	5171880,11	10950	5171880,11
ARKEN25F_130	17	14436,26	21	13867,22	17	13711,55	24	21158,25	30	19810,32	24	19570,30
ARKEN25P_131	234	166971,51	238	158382,56	234	158216,48	334	244794,27	340	226260,80	334	226007,00
MAC_136	23	16651,60	28	16024,15	23	15819,23	32	24371,90	40	22891,65	32	22543,88
RIO25E_143	38	29979,75	42	28590,33	38	28441,86	53	43899,22	60	40843,33	53	40557,59
STE1R_147	15	11234,99	21	10941,50	15	10678,35	22	16502,47	30	15630,71	22	15271,11

4.5 Conclusões

Neste capítulo foram propostos dois modelos de programação inteira para o problema integrado de corte bidimensional e o problema de dimensionamento de lotes. O primeiro modelo proposto é uma extensão do modelo de programação inteira apresentado no capítulo 3 para o problema de corte bidimensional. O segundo modelo é uma extensão do modelo de Dyckhoff (1981) para o problema de corte a uma dimensão.

Nos modelos apresentados é balanceada a antecipação da produção dos itens em cada período, com o intuito de minimizar os custos do processo de corte, os custos de armazenamento e os custos das placas iniciais, respeitando a procura de cada tipo de item em cada período do horizonte de planeamento.

Considerando os resultados computacionais obtidos, pode concluir-se que os modelos I e II são muito semelhantes. As instâncias reais foram resolvidas otimamente em tempo computacional aceitável e comparando estas soluções com as soluções heurísticas, observa-se que, quando o número de períodos aumenta, os ganhos obtidos pelas soluções ótimas também aumentam.

Os modelos propostos foram desenvolvidos para uma variante do problema *2CS-LSP* na qual é permitida a antecipação da produção dos itens e o armazenamento de sobras, com o objetivo de minimizar os custos de produção e de armazenamento. Na literatura esta característica nunca foi abordada, contudo existem alguns problemas similares. Além de ter sido apresentada uma abordagem que pode permitir obter melhores resultados em termos de utilização da matéria-prima, é importante salientar que foram obtidas soluções exatas para o modelo proposto, enquanto nos trabalhos anteriores foram apenas obtidos resultados heurísticos para instâncias de tamanho similar.

Capítulo 5

Problema de empacotamento

Neste capítulo é proposta uma aplicação do *framework* SearchCol (pesquisa meta-heurística por geração de colunas), que combina geração de colunas com pesquisa meta-heurística, ao problema de empacotamento bidimensional. Propõe-se um algoritmo de geração de colunas onde as variáveis do problema mestre restrito correspondem a uma seleção de padrões de empacotamento agrupados pelo item que está posicionado no canto inferior esquerdo (inicializa o padrão). Cada subproblema propõe padrões de empacotamento inicializados por um item diferente e os subproblemas são resolvidos através de uma adaptação da abordagem clássica de Gilmore & Gomory (1965) para o problema de corte bidimensional com padrões do tipo dois-estágios não-exato. Foram realizados testes computacionais com instâncias da literatura e os resultados obtidos com a aplicação de um algoritmo SearchCol ao problema de empacotamento bidimensional são comparados com um modelo de afetação, com o modelo de programação inteira proposto no capítulo 3 e com heurísticas e meta-heurísticas da literatura.

5.1 Introdução

O problema de empacotamento bidimensional é um problema de otimização combinatória com o qual muitas indústrias têm que lidar (por exemplo: madeira e vidro). A principal diferença entre os problemas de corte e os problemas de empacotamento é a heterogeneidade dos itens, sendo que nos problemas de empacotamento a procura é maioritariamente igual a um. No capítulo 3 foram apresentados testes computacionais para algumas instâncias do problema de empacotamento, no entanto pretende-se abordar instâncias de maior dimensão para este problema. Além disso, o desempenho do modelo de programação inteira proposto no capítulo 3 é pior para problemas de empacotamento do que para problemas de corte. Deste modo, é abordado, neste capítulo, o problema de empacotamento bidimensional.

O capítulo inicia-se com uma descrição do *framework* SearchCol. Na secção 5.3 é proposto um novo modelo de decomposição para o problema de empacotamento, apresenta-se o problema mestre e os subproblemas. Na secção pesquisa meta-heurística são descritas as estratégias de pesquisa utilizadas para a obtenção de soluções inteiras. A secção 5.5 é dedicada às perturbações realizadas na geração de colunas. Os testes computacionais foram realizados em dois conjuntos de instâncias da literatura e são apresentados na secção 5.6. Por último, na secção 5.7 são apresentadas as conclusões finais deste capítulo.

5.2 SearchCol

A pesquisa meta-heurística por geração de colunas, SearchCol (*Search by Column Generation*) foi proposta por Alvelos et al. (2010) e define-se como um método otimização que combina geração de colunas com meta-heurísticas, de modo a

obter soluções de boa qualidade e rapidamente, para problemas de otimização combinatória decomponíveis.

O SearchCol assenta no conceito de que o contexto de decomposição fornecido pela geração de colunas constitui uma base poderosa para a definição de meta-heurísticas, como se verificará ao longo deste capítulo.

No SearchCol é tomada a perspetiva combinatória do problema a resolver, nesta ótica, a solução para um problema é vista como uma combinação de soluções de (sub)problemas mais pequenos, que são gerados pelo algoritmo de geração de colunas. Deste modo, em vez do recurso a estratégias de enumeração implícita (por exemplo *branch-and-price*) para resolver o problema mestre inteiro, o problema é abordado como a seleção de uma solução de cada subproblema.

Um elemento fundamental do SearchCol é obviamente a geração de colunas, um método de decomposição guiado por programação linear. A geração de colunas é utilizada para definir o espaço restrito de pesquisa, mas também para guiar a pesquisa das meta-heurísticas.

De um modo geral, o SearchCol divide-se em três grandes fases. No primeiro passo é aplicada a geração de colunas, obtendo-se a solução ótima para a relaxação linear do problema inteiro e as soluções dos subproblemas (associados com as colunas do problema mestre restrito), obtendo-se assim também um limite inferior para o valor da solução ótima inteira. Este limite inferior é útil para superar duas desvantagens usuais das meta-heurísticas: a falta de uma medida da qualidade das soluções e a falta de um critério de paragem bem definido. O SearchCol obtém uma solução e estima a sua distância à solução ótima, e termina se encontrar uma solução com boa qualidade, sendo que, a qualidade é medida através do intervalo entre o valor da solução e o limite inferior.

Na segunda fase do algoritmo SearchCol a pesquisa meta-heurística é conduzida no espaço restrito de pesquisa fornecido pela geração de colunas, isto é

considerando apenas as soluções dos subproblemas gerados pela geração de colunas.

Na última fase uma perturbação no algoritmo de geração de colunas é introduzida, sendo em seguida iniciada uma nova iteração do SearchCol. Uma perturbação é um conjunto adicional de restrições definidas nas variáveis dos subproblemas que são adicionadas ao problema mestre restrito fixando as variáveis dos subproblemas a zero ou um.

Salienta-se que a ordem de execução da geração de colunas e das meta-heurísticas de pesquisa é efetuada de um modo intercalado, sendo a sua combinação obtida colaborativamente através da troca de informações entre ambas, não sendo, no entanto, nenhuma subordinada da outra.

Apresenta-se a seguir uma versão simples do algoritmo SearchCol, um versão mais pormenorizada é dada em Alvelos et al. (2010).

Algoritmo 5: Versão simplificada do algoritmo SearchCol

enquanto *Critério de paragem não satisfeito* **faça**
    Aplicar a geração de colunas;
    Gerar uma solução inteira;
    Aplicar uma meta-heurística;
    Perturbar a geração de colunas;
fim

Em cada iteração do algoritmo SearchCol a geração de colunas fornece a solução ótima, primal e dual para a pesquisa meta-heurística, que por sua vez devolve informação que pode ser utilizada para perturbar a geração de colunas e para começar uma nova iteração.

Os critérios de paragem utilizados no SearchCol são três: (i) intervalo relativo ($|Z_{inc} - Z_{LR}|/|Z_{inc}|$) onde Z_{inc} representa o valor da solução incumbente e Z_{LR} o valor da relaxação linear, que tem de ser inferior a um dado valor, (ii) um limite

para o número de iterações sem melhoria da solução incumbente e (iii) um tempo máximo.

Nesta tese foi utilizado o *framework* SearchCol++. Este *framework* possibilita múltiplas alternativas para diferentes problemas. Neste capítulo apenas se referem as utilizadas neste trabalho. Em anexo, apresentam-se os resultados de afinação de parâmetros de cada algoritmo.

5.3 Modelo de decomposição

5.3.1 Problema Mestre

No problema de empacotamento a duas dimensões (2BP), tal como definido no capítulo 2, um número virtualmente infinito de placas retangulares idênticas (com o mesmo tamanho) está disponível para empacotar um conjunto de itens retangulares. Os itens têm tamanhos muito heterogéneos, sendo a sua procura maioritariamente igual a um.

No problema de 2BP considerado o que se pretende é empacotar todos os itens no menor número de placas possível. Na variante aqui abordada os itens devem ser empacotados nas placas em níveis tal como nos padrões de corte de dois-estágios no problema de corte a duas dimensões do capítulo 3. Seguindo a tipologia de Wäscher et al. (2007) trata-se do problema de empacotamento a duas dimensões com placas retangulares únicas (SBSBPP).

Considere-se o seguinte modelo de programação inteira:

$$(2BP) \quad \text{Minimizar} \quad \sum_{k \in K} \sum_{j \in J^k} y_j^k \tag{5.1}$$

Sujeito a:

$$y_0^k + \sum_{j \in J^k} y_j^k = 1 \qquad k \in K = \{1, ..., n\} \quad (5.2)$$

$$\sum_{j \in J^k} a_{ij}^k y_j^k \geq 1 \qquad i \in I = \{1, ..., n\} \quad (5.3)$$

$$y_j^k \in \{0, 1\} \qquad k \in K, j \in J^k \quad (5.4)$$

$$y_0^k \in \{0, 1\} \qquad k \in K \quad (5.5)$$

Neste modelo o conjunto das variáveis de decisão é dividido em n subconjuntos indexados por k. Para cada subconjunto k, o conjunto dos índices das variáveis de decisão correspondentes é representado por J^k.

As variáveis de decisão estão associados a um possível empacotamento de um subconjunto de itens numa única placa.

A variável de decisão y_j^k é igual a 1 se o padrão de empacotamento j associado ao empacotamento de um subconjunto de itens numa placa iniciada pelo item k (item k posicionado no canto inferior esquerdo) é utilizado e 0 caso contrário. Sendo assim, $y_j^k,\ \ k \in K, j \in J^k$ representa o j-ésimo padrão de empacotamento no qual a placa é iniciada pelo item k. Consideram-se também as variáveis binárias $y_0^k \in \{0, 1\}\ \ k \in K$ que representam uma placa vazia associada a cada item k.

A função objetivo (5.1) traduz o que se pretende, minimizar o número de placas utilizadas, considerando que cada variável representa uma combinação de itens numa única placa iniciada por um item k.

O primeiro conjunto de restrições (5.2), uma restrição para cada item $k, k = 1, ..., n$, garantem que é escolhido um padrão de empacotamento iniciado por cada item, incluindo o padrão de empacotamento com a placa vazia (y_0^k), estas restrições são redundantes, no entanto representam a estrutura combinatória do problema.

O segundo conjunto de restrições (5.3) garantem que todos os itens são em-

pacotados, sendo por isso considerada uma restrição para cada item. Nestas restrições o coeficiente a_{ij}^k é igual a 1 se o item i está empacotado no padrão de empacotamento j iniciado pelo item k e igual a 0 caso contrário.

De um modo genérico pode dizer-se que o modelo traduz a minimização do número de padrões de empacotamento, iniciados por diferentes itens, garantindo que são empacotados todos os itens necessários.

Considera-se que os itens estão indexados por i e estão por ordem decrescente de alturas, $h_1 \geq h_2 \geq h_3 \geq ... \geq h_n$; além disso cada item inicializa uma placa ou é empacotado num nível iniciado por um item com menor índice. Seguindo a mesma linha de raciocínio, o nível inicializado pelo k-ésimo item inicia uma placa ou é empacotado numa placa com menor índice.

Exemplo 5.1 Na figura 5.1 são apresentados exemplos de padrões de empacotamento admissíveis para cada subproblema de um problema de 2BP com sete itens.

No exemplo para $k = 6$ são apresentados todos os padrões de empacotamento possíveis, em y_1^6 é apenas empacotado o item 6 e para y_2^6 são empacotados os itens 6 e 7. Nenhum outro item poderia ser empacotado neste subproblema, uma vez que são apenas considerados os itens com altura maior ou igual à altura do item 6. Cada subproblema k propõe um padrão inicializado pelo item k ao problema mestre.

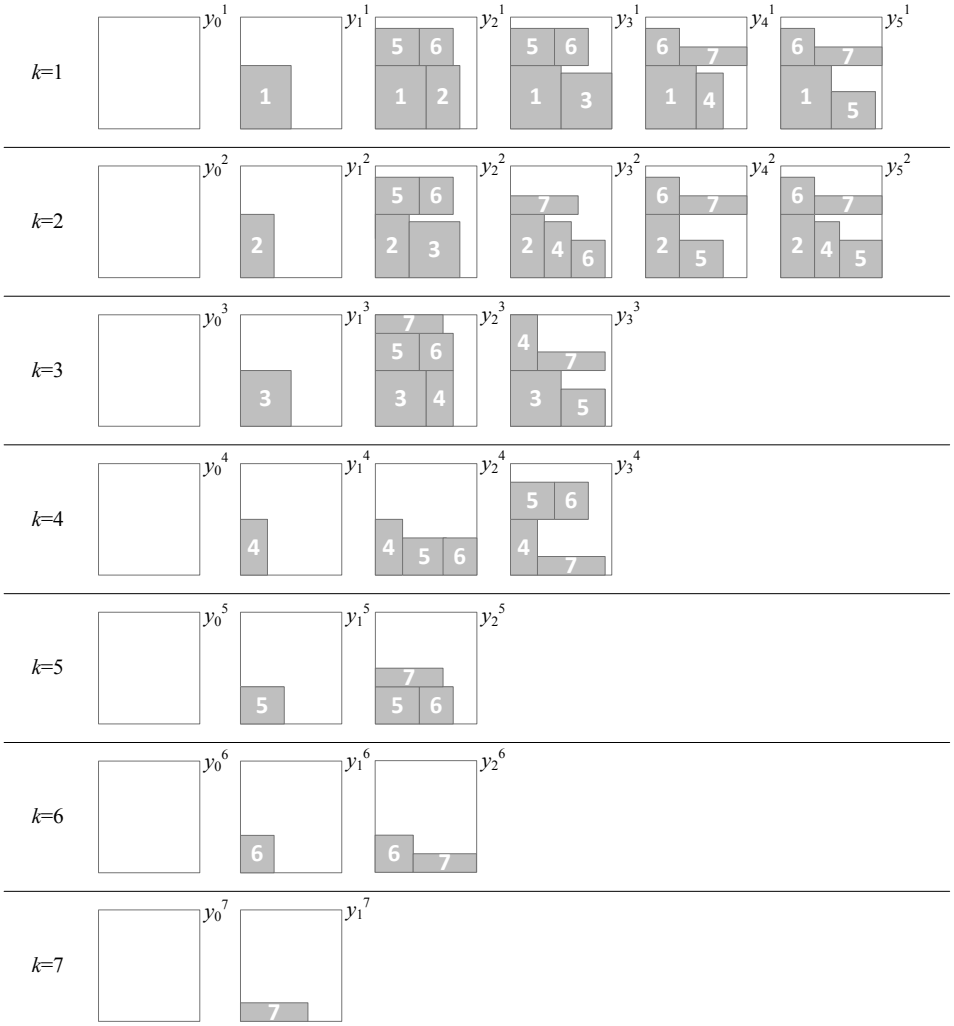


Figura 5.1: Exemplos de padrões de corte obtidos nos diferentes subproblemas.

Na figura 5.2 é ilustrada uma solução que satisfaz a disposição descrita anteriormente para a disposição dos itens nas placas. Os níveis iniciados pelos itens 2, 4, 7 e 9 e as placas iniciadas pelos itens 2, 4, 5, 6, 7, 8 e 9 não são utilizados na solução. A placa à direita é iniciada pelo nível iniciado pelo item 3 e contém ainda os níveis iniciados pelos itens 5 6.

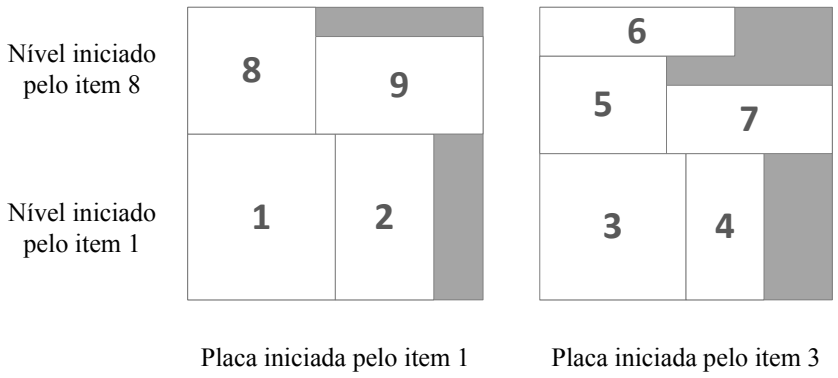


Figura 5.2: Exemplo de uma solução possível para o problema 2BP.

O modelo de programação inteira (5.1)-(5.5) é de difícil resolução devido ao número elevado de padrões de empacotamento possíveis, isto é, devido ao número elevado de variáveis de decisão. A enumeração de todos estes padrões de empacotamento (variáveis de decisão) é inviável. De modo a contornar esta dificuldade recorremos à geração de colunas, que permite lidar com este grande número de varáveis.

O problema mestre é definido pelo modelo (5.1)-(5.5) com as condições de integralidade (5.4)-(5.5) relaxadas e é resolvido através do método de geração de colunas. Uma vez que não são utilizados todos os padrões de empacotamento possíveis (variáveis de decisão) no problema mestre, apenas um conjunto de variáveis de decisão promissoras, a inclusão de variáveis é efetuada de uma forma iterativa, a partir de um espaço de soluções restrito. Este problema é denominado de problema mestre restrito.

Em cada iteração da geração de colunas um problema mestre restrito é resolvido, fornecendo o valor das variáveis duais aos subproblemas, que com esta informação dual geram variáveis (colunas) promissoras a inserir no problema mes-

tre restrito para a próxima iteração.

Os subproblemas consistem então na criação de padrões de empacotamento iniciados pelos diferentes itens. Apresenta-se em seguida o método utilizado para resolver os subproblemas.

5.3.2 Subproblemas

Os subproblemas são diferentes, o subproblema k está associado à placa que é iniciada pelo nível iniciado pelo item k.

Os subproblemas são resolvidos por uma modificação na abordagem clássica de Gilmore & Gomory (1965), para o problema de corte a duas dimensões com padrões de corte do tipo dois estágio não-exato.

Para o subproblema k, é resolvido um problema de mochila para cada nível l, $l \geq k$, considerando todos os itens i tais que $i \geq l$.

O lucro de um item é igual ao valor da variável dual da i-ésima restrição de ligação (ω_i), o peso do item i é a sua largura (w_i) e o volume da mochila é a largura da placa menos a largura do item l que inicia o nível (w_l).

$$\text{Maximizar } U_l = \omega_l + \sum_{a=l+1}^{n} \omega_a b_a \tag{5.6}$$

$$\text{Sujeito a:} \quad \sum_{a=l+1}^{n} w_a b_a \leq W - w_l \tag{5.7}$$

$$b_a \in \{0, 1\} \tag{5.8}$$

O valor ótimo do problema de mochila para cada nível (U_l) é utilizado como lucro noutro problema de mochila em que se decide quais os níveis que devem ser empacotados na placa. Os pesos são dados pelas alturas dos níveis, que estão associadas às alturas dos itens que os iniciam, e o valor do volume resulta da diferença entre a altura da placa e a altura do item k, uma vez que o subproblema

k fornece ao problema mestre padrões de empacotamento onde o item k inicializa a placa.

$$\text{Maximizar } Z_{SP^k} = \quad U_k + \sum_{a=k+1}^{n} U_a x_a \tag{5.9}$$

$$\text{Sujeito a:} \qquad \sum_{a=k+1}^{n} h_a x_a \leq H - h_k \tag{5.10}$$

$$x_a \in \{0, 1\} \tag{5.11}$$

O algoritmo geral para a resolução de cada subproblema é apresentado no algoritmo 6.

Algoritmo 6: Algoritmo para resolver os subproblemas.

```
início
    para todo o subproblema k, k = 1, ..., n faça
        para todo o nível l, l = k, ..., n faça
            Calcular U_l, o valor do nível ótimo iniciado pelo item l;
        fim
        Calcular Z_SP^k, o valor ótimo do padrão de empacotamento iniciado pelo item k;
    fim
fim
```

Uma vez definidos todos os elementos da GC para o problema 2BP, apresenta-se em seguida o algoritmo de GC.

Algoritmo 7: Algoritmo de geração de colunas.

início
- Inicializar o problema mestre restrito e os subproblemas $SP^k, k \in K$;
- **repita**
  - Otimizar o problema mestre restrito;
  - Obter os valores duais ótimos de $\pi^k, k \in K$ e $\omega_i, i \in I$;
  - fim = verdade;
  - **para todo** $k \in K$ **faça**
    - Modificar a função objetivo do subproblema SP^k de acordo com π^k e ω_i;
    - Otimizar o subproblema SP^k;
    - Z_{SP^k} = valor ótimo do SP^k;
    - **se** $Z_{SP^k} < 0$ **então**
      - Obter o problema mestre restrito associado com a solução ótima do SP^k;
      - Atualizar o problema mestre restrito adicionando a nova coluna;
      - fim = falso;
    - **fim**
  - **fim**
- **até** *até fim = verdade*;

fim

Os modelos apresentados em (5.6)-(5.11) são muito semelhantes aos apresentados em (2.4)-(2.10) no capítulo 2 para o modelo de Gilmore & Gomory (1965), a única diferença está no tipo de variáveis, que passam de inteiras para binárias.

Com a resolução dos subproblemas de modo apresentado anteriormente, no final poderemos ter um padrão de empacotamento em que um item esteja empacotado mais do que uma vez, pois pode ser selecionado em diferentes níveis. Deste modo, a coluna a inserir no problema mestre restrito pode ser constituída por valores maiores do que 1. No entanto, considerou-se no modelo que a_{ij}^k é igual a 1 se o item está empacotado no padrão de empacotamento j iniciado pelo item k e igual a zero caso contrário. Deste modo, obtém-se, em geral, um limite

inferior melhor e o espaço de pesquisa é menor.

Dado que o subproblema determina uma solução em que um item pode estar empacotado mais do que uma vez, ou seja, o número de de itens i no padrão de empacotamento j proposto pelo subproblema k for maior do que um, torna-se necessário converter esta solução numa solução em que cada item seja empacotado apenas uma vez, mantendo-se assim a eficiência do algoritmo usado no subproblema. Para tal existem duas possibilidades.

A primeira possibilidade consiste em, depois de resolvido o subproblema, passar para 1 os valores da coluna a inserir no problema mestre restrito, se ela tiver valores maiores que 1 para algum item. No entanto, esta alternativa pode não permitir a convergência da geração de colunas, porque para uma dada coluna binária pode existir uma coluna com valores inteiros com os mesmos itens que seja ainda mais atrativa, porque $1 - \sum_i \omega_i X_i$ é tanto menor quanto maior for X_i.

A segunda possibilidade consiste em, depois de calculada a coluna a inserir, se ela tiver valores maiores que 1 para algum item, recalcular o custo reduzido da coluna se estes valores passarem a ser 1. Se a coluna se mantiver atrativa é inserida no problema mestre restrito, caso contrário não é inserida. Esta foi a alternativa adotada para a resolução da geração de colunas para o problema 2BP.

5.4 Pesquisa meta-heurística

A primeira etapa do SearchCol consiste em resolver a relaxação linear do problema 2BP através da geração de colunas, estando no final disponível um problema mestre restrito constituído por um conjunto de colunas que poderão contribuir para obter soluções inteiras de boa qualidade.

Terminada esta etapa inicia-se a pesquisa meta-heurística, utilizando informações fornecidas pelo problema mestre restrito e pela solução incumbente atual.

Com o término da pesquisa meta-heurística dá-se início à fase de perturbação da geração de colunas e por conseguinte a uma nova iteração do algoritmo.

5.4.1 Representação e avaliação das soluções

Como referido anteriormente, o SearchCol adota uma perspetiva combinatória do problema a resolver, pelo que no problema de 2BP o que se pretende é selecionar um elemento de cada subconjunto $J^k, k \in K$, isto é, um padrão de empacotamento em que a placa é iniciada pelo item k, de modo a que o número de placas utilizadas seja mínimo e a procura dos itens seja satisfeita. A pesquisa meta-heurística é conduzida num espaço restrito considerando apenas as soluções dos subproblemas gerados pela geração de colunas.

Tendo por base esta estrutura combinatória, uma solução pode ser representada por $S = (s(1), s(2), ..., s(n))$, onde $s(k), k \in K$, representa o índice da coluna do subproblema k incluído na solução global S. Esta representação da solução será utilizada pelas meta-heurísticas que serão apresentadas nas subsecções seguintes.

Exemplo 5.2 Para um problema de 2BP com sete subproblemas (um para cada item), se tal como no exemplo da figura 5.1 forem geradas 4 colunas para o primeiro e segundo subproblema, 3 colunas para o subproblema três e quatro, 2 colunas para os subproblemas cinco e seis e 1 coluna para o subproblema sete, existem $5 \times 5 \times 4 \times 4 \times 3 \times 3 \times 2 = 576$ soluções possíveis (incluindo as placas vazias) no espaço restrito definido por estas colunas.

Exemplo de soluções:

$$S^1 = (0, 0, 0, 0, 0, 0, 0)$$

$$S^2 = (0, 0, 0, 0, 0, 0, 1)$$

$$S^3 = (0, 0, 0, 0, 0, 1, 0)$$

$$S^4 = (0, 0, 0, 0, 0, 2, 0)$$

$$S^5 = (0, 0, 0, 0, 1, 0, 0)$$

$$S^6 = (0, 0, 0, 0, 2, 0, 0)$$

$$S^7 = (0, 0, 0, 1, 0, 0, 0)$$

$$S^8 = (0, 0, 0, 2, 0, 0, 0)$$

$$S^9 = (0, 0, 0, 3, 0, 0, 0)$$

...

Considerando novamente o problema de 2BP com sete subproblemas apresentado na figura 5.1, uma solução admissível para este problema é a solução S' apresentada na figura 5.3. Esta solução é constituída pelo quinto padrão gerado pelo subproblema 1, pelo segundo padrão proposto pelo subproblema 2, pelo padrão vazio associado ao subproblema 3, pelo primeiro padrão de empacotamento fornecido pelo subproblema 4 e pelas placas vazias para os restantes subproblemas.

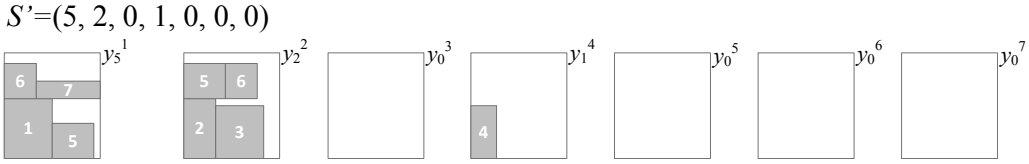


Figura 5.3: Representação de uma solução admissível.

Ao longo das iterações do SearchCol é necessário comparar soluções. Para esse efeito assumiu-se que uma solução admissível é sempre melhor que uma solução

não admissível, que se as duas soluções a avaliar são admissíveis então a melhor é a que utiliza menos placas. Por outro lado se as soluções são não admissíveis, então a melhor é a que viola menos restrições, ou seja, a solução que garante a procura de mais itens.

5.4.2 Estruturas de vizinhança

No algoritmo SearchCol desenvolvido são utilizadas três estruturas de vizinhança.

Na vizinhança 1 duas soluções são consideradas vizinhas se entre elas existir apenas uma modificação, isto é, se entre duas soluções apenas um índice de uma coluna associada a um subproblema k (padrão de empacotamento inicializado pelo item k) é diferente.

Para a vizinhança 2 duas soluções são consideradas vizinhas se entre elas existirem no máximo duas modificações, isto é, no máximo dois índices associados a dois subproblemas são diferentes. Na vizinhança 3 são permitidas no máximo três índices diferentes associados aos subproblemas.

Na figura 5.4 estão representadas 3 soluções (a, b e c) para o exemplo apresentado na figura 5.1. A solução S^a é vizinha de S^b na estrutura de vizinhança 1, uma vez que entre uma e outra apenas o subproblema um tem solução diferente. As soluções S^a e S^c são vizinhas na estrutura de vizinhança 2, havendo diferenças nos índices das soluções dos subproblemas dois e três. É de salientar que as soluções S^a e S^b são também vizinhas na vizinhança 2, porque a vizinhança 2 engloba a vizinhança 1.

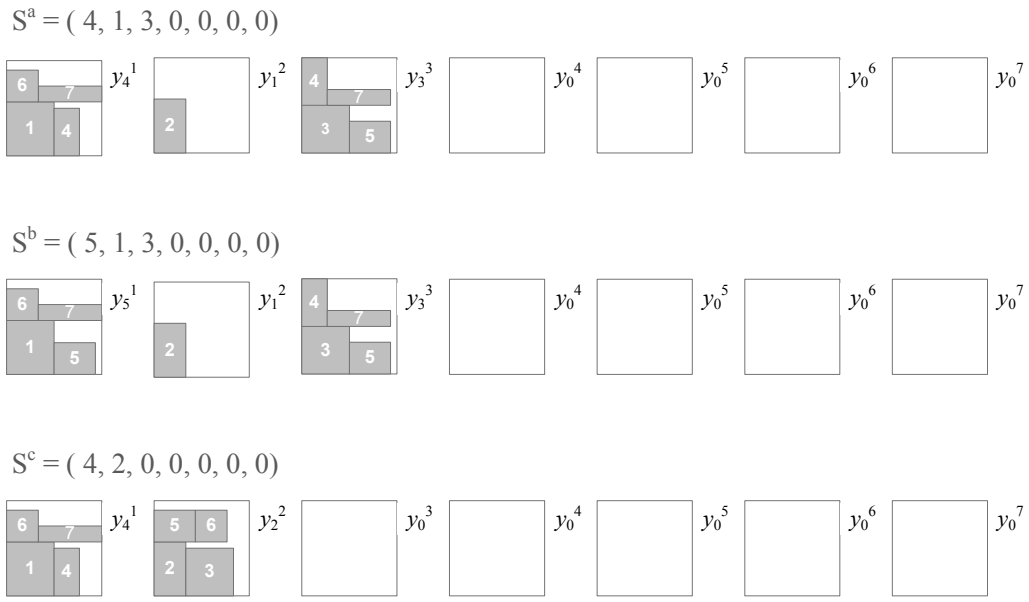


Figura 5.4: Exemplos de soluções vizinhas nas vizinhanças 1 e 2.

As meta-heurísticas pesquisa local multi-início, pesquisa em vizinhanças variáveis e Mipheur foram utilizadas na fase de pesquisa, apresenta-se em seguida uma descrição das mesmas.

5.4.3 Pesquisa local multi-início

Os algoritmos de pesquisa local são construídos como uma forma de exploração do espaço de pesquisa. Parte-se de uma solução inicial (usualmente obtida através de um método construtivo) e em cada iteração procura-se a melhoria da solução atual através de uma pesquisa na sua vizinhança. O processo termina quando nenhuma solução vizinha é melhor do que a atual.

Existem dois tipos de pesquisa local: a primeira melhoria (*first improvement*) e a melhor descida (*best improvement*). Na primeira melhoria em cada iteração

seleciona-se a primeira solução encontrada cujo valor da função é menor que o valor da solução atual. Enquanto na melhor descida são analisados todos os vizinhos e apenas o melhor de todos é selecionado. Na aplicação do SearchCol ao problema 2BP abordado neste trabalho foi utilizada a pesquisa local com primeira melhoria.

No procedimento de pesquisa local adotado para a aplicação do *framework* SearchCol foi utilizada a estrutura de vizinhança 1.

A pesquisa local multi-início (*Multi-start Local Search*) consiste em repetir a pesquisa local a partir de soluções iniciais diferentes. No SearchCol estão implementados diferentes métodos para construir as soluções iniciais, assim como um parâmetro que controla a aleatoriedade na construção de uma solução. Na primeira pesquisa local da pesquisa multi-início a solução inicial é construída selecionando a primeira solução gerada por cada subproblema na fase de geração de colunas. Enquanto que, para as restantes pesquisas locais a solução inicial é construída selecionando-se aleatoriamente uma solução de cada subproblema, atribui-se a cada variável associada a cada subproblema uma probabilidade que depende da solução incumbente ser ou não atualizada na pesquisa local anterior. Se a solução incumbente foi atualizada na pesquisa local anterior, é atribuída a cada variável o seu valor na solução ótima do problema mestre restrito; se pelo contrário, não existiu uma alteração na solução incumbente, a probabilidade de uma variável ser selecionada segue uma distribuição uniforme.

Na pesquisa local multi-início foi definido o máximo de uma iteração sem melhoria da solução.

5.4.4 Pesquisa em vizinhanças variáveis

A meta-heurística pesquisa em vizinhanças variáveis mais conhecida por VNS (*Variable Neighborhood Search*) foi proposta por Mladenovic & Hansen (1997) e

consiste em explorar o espaço de soluções através de mudanças sistemáticas das estruturas de vizinhança da pesquisa local.

A VNS vai explorando vizinhanças cada vez mais distantes da solução incumbente atual. O algoritmo da meta-heurística pesquisa em vizinhanças variáveis é apresentado no algoritmo 8.

Algoritmo 8: Algoritmo da pesquisa em vizinhanças variáveis.

início
  Selecionar o conjunto de estruturas de vizinhança $N_l, l = 1, ..., l_{max}$ que serão utilizadas na pesquisa;
  Encontrar uma solução inicial s;
  Definir um critério de paragem ;
  l=1;
  repita
    Gerar aleatoriamente um ponto s^* da l-ésima vizinhança de s ($s^* \in N_l(s)$);
    Aplicar um método de pesquisa local considerando s^* como solução inicial;
    Denominar o ótimo local encontrado por s^{**} ;
    se *s^{**} é melhor que a solução incumbente* **então**
      $s = s^{**}$;
      Continuar a a pesquisa local com $N_1(l = 1)$;
    senão
      $l = l + 1$;
    fim
  até *até* $l = l_{max}$;
fim

A VNS começa com uma solução inicial s e em cada iteração é selecionado aleatoriamente um vizinho s^* na vizinhança $N_l(s)$ da solução s atual. É aplicada a pesquisa local à solução vizinha s^*, obtendo-se no final deste procedimento o ótimo local s^{**}. Se o valor desta solução for melhor do que o valor da solução atual s, a pesquisa local continua recomeçando da solução s^{**} e da primeira estrutura de

vizinhança $N_1(s)$. Caso contrário, a pesquisa local continua a partir da próxima estrutura de vizinhança. O algoritmo termina quando é atingido um critério de paragem pré-estabelecido.

Para o problema 2BP as estruturas de vizinhança utilizadas são as três estruturas de vizinhança apresentadas anteriormente. Sendo assim, o número máximo de estruturas de vizinhança é 3 ($l_{max} = 3$).

Tal como na pesquisa local multi-início, a solução inicial é construída selecionando a primeira solução gerada por cada subproblema.

A estratégia de pesquisa local adotada é novamente a primeira melhoria, ou seja, em cada iteração seleciona-se a primeira solução encontrada cujo valor é menor que o valor da solução atual.

5.4.5 Pesquisa Mipheur

Na meta-heurística Mipheur depois de resolvida a relaxação linear, a fase de pesquisa meta-heurística consiste em resolver de forma exata um problema de programação inteira com as colunas que constituem o problema mestre restrito.

5.5 Perturbações

No final da fase de pesquisa meta-heurística (pesquisa local multi-início ou VNS) existe uma solução incumbente, esta solução é utilizada para perturbar a geração de colunas na iteração seguinte do algoritmo SearchCol.

O que se pretende é fazer alterações na geração de colunas de modo a que sejam geradas soluções nos subproblemas (padrões de empacotamento) que permitam melhorar a solução incumbente.

Uma perturbação é um conjunto adicional de restrições definidas nas variáveis dos subproblemas que são adicionadas ao problema mestre restrito fixando as

variáveis dos subproblemas a zero ou um.

Na aplicação do SearchCol ao problema 2BP foram utilizadas dois tipos de perturbações que estão relacionadas com o facto da solução incumbente ser ou não atualizada na iteração anterior. A perturbação baseada na solução incumbente é realizada sempre que a solução incumbente foi atualizada, enquanto que a perturbação baseada na diferença entre a solução ótima da geração de colunas e a solução incumbente é utilizada quando a solução incumbente não é melhorada.

Na perturbação baseada na solução incumbente selecionam-se aleatoriamente variáveis dos subproblemas (a quantidade de variáveis a escolher é definida por um parâmetro) do conjunto das variáveis do problema mestre restrito que pertencem à solução incumbente. Se for selecionada a variável $x_i^k = 1$ (o item i está empacotado na placa iniciada pelo item k) é adicionada uma restrição no problema mestre restrito que obriga à seleção de um padrão de empacotamento, que seja proposto pelo subproblema k e que contenha o item i. A nova restrição terá coeficiente 1 em todas as colunas/variáveis fornecidas pelo subproblema k que tenham coeficiente 1 na linha i e será maior ou igual a 1. Se a variável selecionada for $x_i^k = 0$ (o item i não está empacotado na placa iniciada pelo item k) força-se que não sejam escolhidos padrões de empacotamento que sejam iniciados pelo item k e em que o item i também esteja empacotado. A nova restrição terá coeficiente 1 em todas as colunas/variáveis fornecidas pelo subproblema k que tenham coeficiente 1 na linha i e será menor ou igual a 0.

Exemplo 5.3 Considere-se a solução incumbente $S' = (5, 2, 0, 1, 0, 0, 0)$ representada na figura 5.3. Se fosse selecionada aleatoriamente a variável $x_5^1 = 1$ do subproblema 1 (o item 5 está empacotado na placa iniciada pelo item 1), deveria ser adicionada uma restrição no problema mestre restrito que forçasse a que seja utilizado um padrão em que o item 5 esteja empacotado.

Se por outro lado fosse selecionada uma variável de um subproblema com valor

zero, por exemplo a variável $x_4^2 = 0$ (o item 4 não é empacotado na placa iniciada pelo item 2) deveria ser adicionada uma restrição ao problema mestre restrito que obrigasse a que não sejam escolhidos padrões de empacotamento iniciados pelo item 2 em que o item 5 também esteja empacotado.

Na tabela 5.1 está representada a matriz das restrições para o problema mestre restrito com os padrões de empacotamento propostos pelos subproblemas apresentados na figura 5.1 para uma pequena instância com sete itens. As restrições a cinzento representam as perturbações realizadas com base na solução incumbente apresentadas no exemplo anterior.

Tabela 5.1: Matriz das restrições do problema mestre restrito com restrições de perturbação.

	y_0^1	y_1^1	y_2^1	y_3^1	y_4^1	y_5^1	y_0^2	y_1^2	y_2^2	y_3^2	y_4^2	y_5^2	y_0^3	y_1^3	y_2^3	y_3^3	y_0^4	y_1^4	y_2^4	y_3^4	y_0^5	y_1^5	y_2^5	y_0^6	y_1^6	y_2^6	y_0^7	y_1^7	
1		1	1	1	1	1																						$\geq$ 1
2			1					1	1	1	1	1																$\geq$ 1
3				1					1					1	1	1												$\geq$ 1
4					1					1		1			1	1		1	1	1									$\geq$ 1
5			1	1		1			1		1	1			1	1			1	1		1	1						$\geq$ 1
6			1	1	1	1			1	1	1	1			1				1	1			1		1	1			$\geq$ 1
7					1	1				1	1	1			1	1				1			1			1			$\geq$ 1
			1	1		1																							$\geq$ 1
										1		1																	$\leq$ 0

Na perturbação baseada na diferença entre a solução ótima da geração de colunas e a solução incumbente, se para uma variável selecionada ambas as soluções forem iguais a 1 é adicionada uma restrição no problema mestre restrito que obrigue esta variável a ser 1 no subproblema. Se por outro lado a variável selecionada é igual a zero quer na solução ótima da geração de colunas quer na solução incumbente, é adicionada uma restrição no problema mestre restrito que obrigue a variável a ser zero no subproblema.

5.6 Resultados computacionais

SearchCol++ é um *framework* em C++ que implementa algoritmos SearchCol. Para a implementação de um algoritmo SearchCol é necessário programar quatro funções:

- ***loadDecomposition()*** define o número de subproblemas, o número de variáveis de cada subproblema, o sentido das restrições e os termos independentes.
- ***solveSP()*** implementa o algoritmo do subproblema.
- ***iniModifiedObj()*** recebe o valor das variáveis duais e inicializa os coeficientes da função objetivo das variáveis dos subproblemas.
- ***setColofSol()*** recebe a solução do subproblema e devolve uma coluna para o problema mestre restrito.

Diferentes meta-heurísticas de pesquisa e perturbações estão também implementadas no *framework* SearchCol++. Para a aplicação ao problema 2BP foi, como descrito anteriormente, utilizada a meta-heurística pesquisa local multi-início, a pesquisa em vizinhanças variáveis e a pesquisa Mipheur. As perturbações usadas foram a baseada na solução incumbente e a baseada na diferença entre a solução ótima da geração de colunas e a solução incumbente.

Além disso são apresentados testes computacionais para aplicações do SearchCol ao problema 2BP sem a fase de perturbação da geração de colunas. Neste caso, é resolvida a geração de colunas e o algoritmo para depois da pesquisa meta-heurística (PLMI, VNS ou Mipheur).

Para a geração de colunas foi utilizada uma heurística inicial baseada na heurística *Knapsack Packing* de Lodi et al. (1999).

Os itens são ordenados por ordem decrescente de altura e é criada uma lista com os itens ainda não empacotados. Inicia-se um novo nível com o primeiro item da lista e resolve-se um problema de mochila de modo a selecionar os itens que constituirão o nível. A lista é atualizada e repete-se o processo até que todos os itens sejam empacotados. Com todos os itens empacotados em níveis é repetido um processo semelhante, de problemas de mochila, para empacotar os níveis nas placas.

Os resultados da aplicação do SearchCol ao problema de empacotamento bidimensional são comparados com o modelo de programação inteira de Lodi et al. (2004). Este modelo foi construído e implementado em C++, utilizando a biblioteca de funções do *IBM ILOG Cplex 12.1*.

Para os testes computacionais foram utilizados dois conjuntos de instâncias da literatura, o primeiro conjunto é constituído por seis classes propostas por Berkey & Wang (1987) e o segundo conjunto de instâncias é composto por quatro classes apresentadas por Martello & Vigo (1998).

O primeiro conjunto de instâncias tem as seguintes caraterísticas:

Classe 1: h_i and w_i uniformemente aleatória em [1,10], $W = H = 10$

Classe 2: h_i and w_i uniformemente aleatória em [1,10], $W = H = 30$

Classe 3: h_i and w_i uniformemente aleatória em [1, 35], $W = H = 40$

Classe 4: h_i and w_i uniformemente aleatória em [1, 35], $W = H = 100$

Classe 5: h_i and w_i uniformemente aleatória em [1, 100], $W = H = 100$

Classe 6: h_i and w_i uniformemente aleatória em [1, 100], $W = H = 300$

No segundo conjunto de instâncias, $W = H = 100$ e são considerados quatro tipos de itens:

Tipo 1: w_i uniformemente aleatória em [$2/3W$, W], h_i uniformemente aleatória em [1,$1/2H$]

Tipo 2: w_i uniformemente aleatória em [1, $1/2W$], h_i uniformemente aleatória em [$2/3H$,H]

Tipo 3: w_i uniformemente aleatória em [$1/2W$, W], h_i uniformemente aleatória em [$1/2H$,H]

Tipo 4: w_i uniformemente aleatória em [1, $1/2W$], h_i uniformemente aleatória em [1,$1/2H$]

Cada classe foi construída considerando os seguintes parâmetros:

Classe 7: tipo 1 com probabilidade 0.7, tipos 2, 3, e 4 com probabilidade 0.1 cada

Classe 8: tipo 2 com probabilidade 0.7, tipos 1, 3, e 4 com probabilidade 0.1 cada

Classe 9: tipo 3 com probabilidade 0.7, tipos 1, 2, e 4 com probabilidade 0.1 cada

Classe 10: tipo 4 com probabilidade 0.7, tipos 1, 2, e 3 com probabilidade 0.1 cada

Cada classe é constituída por cinco sub-classes, a diferença entre elas é o número total de itens $n = 20, 40, 60, 80, 100$, além disso cada sub-classe é constituída por dez instâncias. Deste modo, no total são testadas 500 instâncias.

Na tabela 5.2 são apresentados os resultados obtidos pelo modelo de programação inteira de Lodi et al. (2004) (ver página 37), para a pesquisa Mipheur sem

perturbações, isto é a geração de colunas é resolvida apenas uma vez e a solução inteira é obtida resolvendo um problema inteiro com as colunas da relaxação linear, e para a pesquisa Mipheur com perturbações.

São apresentados na tabela 5.2, para cada classe a média do valor das soluções de cada subclasse z, o número de soluções ótimas em cada subclasse n_{opt}, o tempo médio para cada subclasse t_{med}, o tempo médio despendido na pesquisa meta-heurística t_{pmh} para cada subclasse e o tempo médio despendido na geração de colunas t_{gc} para cada subclasse. O tempo computacional despendido na perturbação não é apresentado porque é sempre zero.

Tabela 5.2: Resultados para o modelo de Lodi et al. (2004) e para as heurísticas Mipheur com e sem perturbações

Classe	n	Lodi et al. (2004)			Mipheur sem perturbações					Mipheur com perturbações				
		z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	t_{pmh}	t_{gc}	z	n_{opt}	t_{med}	t_{pmh}	t_{gc}
	20	7,3	10	0	7,3	10	0	0	0	7,3	10	0	0	0
	40	13,8	10	0,1	13,9	9	0,1	0	0	13,9	9	0,1	0	0
1	60	20,3	10	0,2	20,4	9	0,2	0	0,1	20,4	9	0,3	0,1	0,2
	80	27,7	10	0,6	27,7	10	0,4	0,1	0,3	27,7	10	0,5	0,2	0,3
	100	32,4	10	2,1	32,5	9	1,3	0,4	0,8	32,4	10	1,9	0,8	1,1
	20	1	10	0	1,1	9	0	0	0	1,1	9	0	0	0
	40	2	10	0,2	2	10	0,3	0	0,3	2	10	0,3	0	0,3
2	60	2,8	9	302	2,8	9	0,7	0	0,6	2,8	9	0,7	0	0,7
	80	3,3	10	36,9	3,4	9	2,6	0	2,5	3,4	9	2,9	0,1	2,8
	100	4,1	9	215,5	4,1	9	5,4	0	5,3	4,1	9	6	0,2	5,8
	20	5,4	10	0	5,5	9	0	0	0	5,5	9	0	0	0
	40	9,8	10	0,2	10	8	0,1	0,1	0,1	9,9	9	0,2	0,1	0,1
3	60	14	10	1,8	14,5	5	0,5	0,3	0,3	14,4	6	0,8	0,5	0,3
	80	19,7	10	8,6	19,7	10	0,8	0,1	0,7	19,7	10	1	0,2	0,8
	100	22,8	10	185,7	23,1	7	3,4	1,8	1,6	23,1	7	5	2,9	2,1
	20	1	10	0	1	10	0	0	0	1	10	0	0	0
	40	2	10	0,4	2	10	0,3	0	0,3	2	10	0,3	0	0,3
4	60	2,6	10	235,6	2,7	9	1,1	0	1	2,7	9	1,1	0	1,1
	80	3,4	9	140	3,4	9	1,3	0	1,3	3,4	9	1,4	0	1,4
	100	4,1	8	939,8	4	9	5,1	0	5	4	9	6,8	0,2	6,5
	20	6,7	10	0	6,7	10	0	0	0	6,7	10	0	0	0
	40	12,3	10	0,1	12,3	10	0,1	0	0,1	12,3	10	0,1	0	0,1
5	60	18,3	10	0,3	18,3	10	0,3	0,1	0,2	18,3	10	0,3	0,1	0,2
	80	25	10	1,8	25,2	8	0,5	0,1	0,4	25,1	9	0,8	0,3	0,4
	100	28,8	10	23,7	29,1	7	1,8	0,6	1,1	29	8	2,7	1,2	1,4
	20	1	10	0	1	10	0	0	0	1	10	0	0	0
6														

Tabela 5.2 Resultados para o modelo de Lodi et al. (2004) e para as heurísticas Mipheur com e sem perturbações *(continuação da página anterior).*

Classe	n	Lodi et al. (2004).			Mipheur sem perturbações					Mipheur com perturbações				
		z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	t_{pmh}	t_{gc}	z	n_{opt}	t_{med}	t_{pmh}	t_{gc}
	40	1,9	10	0,2	1,9	10	0,2	0	0,2	1,9	10	0,3	0	0,2
	60	2,3	9	249,5	2,3	9	1,7	0	1,7	2,3	9	1,8	0	1,7
	80	3	10	3,5	3	10	2,2	0	2,2	3	10	2,3	0	2,2
	100	3,5	9	621,1	3,5	9	5,2	0	5,1	3,5	9	5,5	0,1	5,3
	20	5,7	10	0	5,7	10	0	0	0	5,7	10	0	0	0
	40	11,5	10	0,2	11,7	8	0,1	0	0,1	11,7	8	0,1	0	0,1
7	60	16,2	10	1	16,4	8	0,5	0,2	0,2	16,4	8	0,7	0,4	0,3
	80	23,3	10	3,9	23,3	10	0,9	0,2	0,6	23,3	10	1,3	0,5	0,8
	100	27,5	10	17,9	27,8	7	1,9	0,7	1,1	27,7	8	3,3	1,6	1,7
	20	6,1	10	0	6,1	10	0	0	0	6,1	10	0	0	0
	40	11,5	10	0,5	11,7	8	0,1	0	0,1	11,6	9	0,2	0,1	0,1
8	60	16,4	10	2,3	16,4	10	0,3	0	0,2	16,4	10	0,5	0,1	0,4
	80	22,6	10	11,3	22,6	10	0,8	0,2	0,6	22,6	10	1,4	0,6	0,8
	100	28,1	10	8,1	28,1	10	2,4	1,3	1	28,1	10	3,9	2,3	1,5
	20	14,3	10	0	14,3	10	0	0	0	14,3	10	0	0	0
	40	27,8	10	0	27,8	10	0	0	0	27,8	10	0	0	0
9	60	43,7	10	0	43,7	10	0	0	0	43,7	10	0	0	0
	80	57,7	10	0,1	57,7	10	0,1	0	0	57,7	10	0,1	0	0,1
	100	69,5	10	0,1	69,5	10	0,2	0	0,1	69,5	10	0,2	0	0,2
	20	4,5	10	0	4,6	9	0	0	0	4,6	9	0	0	0
	40	7,7	10	0,4	7,9	8	0,2	0	0,1	7,9	8	0,2	0,1	0,2
10	60	10,5	9	722,7	10,9	5	1,1	0,4	0,7	10,8	6	1,7	0,7	0,9
	80	13,3	9	796,7	13,8	4	3,8	1,6	2,1	13,8	4	6,3	3,7	2,5
	100	16,3	9	808,3	16,9	3	268,9	263,7	5,2	16,6	6	496,6	489,7	6,9
Soma		736,5	490	5343,5	741,3	442	317	271,9	43,5	740,2	453	560	506,8	51,9

Analisando os resultados apresentados na tabela 5.2 conclui-se que o modelo de programação inteira obteve a solução ótima em 98% das instâncias enquanto que as heurísticas Mipheur sem e com perturbações encontraram a solução ótima em 88% e 91% das instâncias respetivamente. No entanto, as heurísticas foram bastante mais rápidas, tendo a Mipheur sem perturbações demorado cerca 17 vezes menos tempo e a heurística Mipheur com perturbações cerca de 10 vezes menos tempo do que o modelo de programação inteira. Nas heurísticas Mipheur a maior parte do tempo de otimização é gasto na pesquisa. Em cada classe

verifica-se um aumento do tempo computacional à medida que o número de itens vai aumentando, sendo que na classe 10 esse aumento é mais evidente.

É de salientar que na classe 4 para a subclasse com $n = 100$ as heurísticas encontraram mais uma solução ótima do que o modelo de programação inteira de Lodi et al. (2004) e em muito menos tempo.

A heurística Mipheur com perturbações, tal como esperado, encontrou melhores soluções do que a heurística sem perturbações, embora o preço dessa melhoria se tenha verificado no aumento do tempo computacional, no entanto estes valores continuam bastante aceitáveis quando comparados com os tempos computacionais do modelo de programação inteira. Além disso, para as instâncias em que o valor da solução não é ótimo, a solução utiliza apenas mais uma placa do que a solução ótima.

Os resultados obtidos com a meta-heurística pesquisa local multi-início (PLMI) sem e com perturbações são apresentados na tabela 5.3. Os testes foram executados 10 vezes para cada instância, devido ao facto da construção da solução inicial ter uma componente aleatória. Para os resultados do SearchCol com as meta-heurísticas PLMI com e sem perturbações apresenta-se para cada subclasse o valor médio das médias das soluções obtidas em todas as execuções (z_{med}), assim como a média de soluções ótimas obtidas para cada subclasse nas 10 execuções (n_{optMed}) e o tempo médio para cada subclasse.

Apresentam-se também novamente os resultados obtidos com a aplicação do SearchCol com as meta-heurísticas Mipheur com e sem perturbações.

Tabela 5.3: Resultados obtidos com o SearchCol com PLMI sem e com perturbações.

Classe	n	Mipheur sem perturbações			Mipheur com perturbações			PLMI sem perturbação			PLMI com perturbações		
		z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	z_{med}	n_{optMed}	t_{med}	z_{med}	n_{optMed}	t_{med}
	20	7,3	10	0	7,3	10	0,0	7,6	8	0,0	7,4	9	0,0
	40	13,9	9	0,1	13,9	9	0,1	14	8	0,1	14	8,1	0,1
1	60	20,4	9	0,2	20,4	9	0,3	21,1	2,3	0,2	21,1	2,4	0,2
	80	27,7	10	0,4	27,7	10	0,5	28,7	2	0,4	28,6	2,7	0,4
	100	32,5	9	1,3	32,4	10	1,9	33,5	0,2	1,0	33,4	0,7	1,3
	20	1,1	9	0	1,1	9	0,0	1,1	9	0,0	1,1	9	0,0
	40	2	10	0,3	2	10	0,3	2	10	0,3	2	10	0,4
2	60	2,8	9	0,7	2,8	9	0,7	2,8	9	0,7	2,8	9	0,9
	80	3,4	9	2,6	3,4	9	2,9	3,4	9	2,4	3,4	9	3,2
	100	4,1	9	5,4	4,1	9	6,0	4,1	9	6,6	4,1	9	7,2
	20	5,5	9	0	5,5	9	0,0	5,6	8,4	0,0	5,6	8,5	0,0
	40	10	8	0,1	9,9	9	0,2	10,3	5	0,1	10,3	5	0,1
3	60	14,5	5	0,5	14,4	6	0,8	15,1	0	0,3	15,1	0,1	0,4
	80	19,7	10	0,8	19,7	10	1,0	20,6	2,2	0,9	20,6	2	1,0
	100	23,1	7	3,4	23,1	7	5,0	24,3	1	2,0	24,3	1	2,5
	20	1	10	0	1	10	0,0	1	10	0,0	1	10	0,0
	40	2	10	0,3	2	10	0,3	2	10	0,3	2	10	0,3
4	60	2,7	9	1,1	2,7	9	1,1	2,7	9	1,2	2,7	9	1,3
	80	3,4	9	1,3	3,4	9	1,4	3,4	9	1,6	3,4	9	1,9
	100	4	9	5,1	4	9	6,8	4	9	6,3	4	9	7,1
	20	6,7	10	0	6,7	10	0,0	7	7,1	0,0	6,8	8,7	0,0
	40	12,3	10	0,1	12,3	10	0,1	12,6	7	0,1	12,6	7,4	0,1
5	60	18,3	10	0,3	18,3	10	0,3	19,1	4	0,2	19	4,5	0,3
	80	25,2	8	0,5	25,1	9	0,8	26,3	1	0,5	26,2	1,5	0,6
	100	29,1	7	1,8	29	8	2,7	30,2	0	1,4	30,1	0,3	1,8
	20	1	10	0	1	10	0,0	1	10	0,0	1	10	0,0
	40	1,9	10	0,2	1,9	10	0,3	1,9	10	0,3	1,9	10	0,3
6	60	2,3	9	1,7	2,3	9	1,8	2,3	9	2,0	2,3	9	2,0
	80	3	10	2,2	3	10	2,3	3	10	2,7	3	10	2,7
	100	3,5	9	5,2	3,5	9	5,5	3,5	9	6,4	3,5	9	6,7
	20	5,7	10	0	5,7	10	0,0	6	7,2	0,0	5,8	8,9	0,0
	40	11,7	8	0,1	11,7	8	0,1	12	5,2	0,1	12	5,5	0,1
7	60	16,4	8	0,5	16,4	8	0,7	17	3	0,3	17	3	0,4
	80	23,3	10	0,9	23,3	10	1,3	24,3	2	0,8	24,3	2	1,1
	100	27,8	7	1,9	27,7	8	3,3	28,5	2	1,5	28,5	2	2,1
	20	6,1	10	0	6,1	10	0,0	6,4	7,1	0,0	6,3	7,9	0,0
	40	11,7	8	0,1	11,6	9	0,2	12,1	5	0,1	12	5,1	0,1
8	60	16,4	10	0,3	16,4	10	0,5	16,8	6	0,3	16,8	6	0,5
	80	22,6	10	0,8	22,6	10	1,4	23,6	2	0,7	23,6	2	1,1
	100	28,1	10	2,4	28,1	10	3,9	29,1	2	1,3	29,1	2	2,0
	20	14,3	10	0	14,3	10	0,0	14,4	9	0,0	14,3	10	0,0
	40	27,8	10	0	27,8	10	0,0	27,9	9	0,0	27,8	10	0,0
9	60	43,7	10	0	43,7	10	0,0	44	7	0,0	43,7	9,7	0,0
	80	57,7	10	0,1	57,7	10	0,1	58	7,3	0,1	57,7	9,7	0,1
	100	69,5	10	0,2	69,5	10	0,2	70	5	0,2	69,6	9,4	0,2
	20	4,6	9	0	4,6	9	0,0	4,6	9	0,0	4,6	9	0,0

10

Tabela 5.3 Resultados obtidos com o SearchCol com PLMI sem e com perturbações *(continuação da página anterior).*

Classe	n	Mipheur sem perturbações			Mipheur com perturbações			PLMI sem perturbações			PLMI com perturbações		
		z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	z_{med}	n_{optMed}	t_{med}	z_{med}	n_{optMed}	t_{med}
	40	7,9	8	0,2	7,9	8	0,2	8,1	6	0,2	8,1	6,1	0,2
	60	10,9	5	1,1	10,8	6	1,7	11,1	3	0,9	11,1	3	1,0
	80	13,8	4	3,8	13,8	4	6,3	14,1	2	2,4	14,1	2	3,0
	100	16,9	3	268,9	16,6	6	496,6	17,2	0	6,1	17,2	0	7,2
Soma		741,3	442	317	740,2	453	560,0	759,2	296	53,1	755,5	316	62,1

Uma análise dos resultados permite concluir que a inserção de perturbações na geração de colunas provoca uma melhoria nos resultados obtidos; a comprovar esta afirmação está o número médio de soluções ótimas que passou de 296 para PLMI sem perturbações para 316 para PLMI com perturbações.

Os tempos computacionais são bastante bons, verificando-se um ligeiro aumento do tempo computacional total quando se utiliza a perturbação na geração de colunas. Os tempos computacionais da geração de colunas não são apresentados, mas estes valores são bastante pequenos, para a meta-heurística PLMI sem perturbações o tempo médio máximo dispendido numa subclasse para a geração de colunas foi de 1,2 segundos e a soma das médias de todas as subclasses foi de 43,1 segundos. Enquanto que para a meta-heurística PLMI com perturbações a soma dos tempos médios de cada subclasse para a geração de colunas foi de 49,9 segundos.

As aplicações do SearchCol com a meta-heurística PLMI sem e com perturbações obtiveram soluções de qualidade inferior quando comparadas com as soluções obtidas com a aplicação do Searchcol com a meta-heurística Mipheur sem e com perturbações. No entanto as primeiras foram bastante mais rápidas, uma vez que a soma dos tempos médios das meta-heurísticas Mipheur são da ordem das

centenas enquanto os tempos médios das meta-heurísticas PLMI são da ordem das dezenas.

Para a aplicação do SearchCol com a meta-heurística VNS sem e com perturbações os resultados computacionais são apresentados na tabela 5.4. Para a aplicação com perturbação os testes computacionais foram repetidos 10 vezes, devido à componente aleatória das perturbações utilizadas.

Tal como na tabela de resultados apresentada para o SearchCol com as meta-heurísticas PLMI incluindo ou não perturbações, na tabela 5.4 apresenta-se para o SearchCol com VNS com perturbações o valor médio das médias das soluções obtidas em todas as execuções (z_{med}), assim como a média de soluções ótimas obtidas para cada subclasse nas 10 execuções (n_{optMed}) e o tempo médio para cada subclasse.

Para as aplicações do SearchCol com PLMI com perturbações e Mipheur sem e com perturbações apresenta-se para cada subclasse o valor médio das soluções, o número de soluções ótimas e o tempo médio.

Tabela 5.4: Resultados obtidos com o SearchCol com VNS sem e com perturbação.

Classe	n	Mipheur sem perturbação			Mipheur com perturbação			VNS sem perturbação			VNS com perturbação		
		z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	z_{med}	n_{optMed}	t_{med}
1	20	7,3	10	0	7,3	10	0,0	7,8	6	0,0	7,75	6,3	0,0
	40	13,9	9	0,1	13,9	9	0,1	14	8	0,1	14	8	0,1
	60	20,4	9	0,2	20,4	9	0,3	21,2	2	0,2	21,2	2	0,2
	80	27,7	10	0,4	27,7	10	0,5	28,7	2	0,4	28,7	2	0,5
	100	32,5	9	1,3	32,4	10	1,9	33,5	0	1,0	33,5	0	1,3
2	20	1,1	9	0	1,1	9	0,0	1,1	9	0,0	1,1	9	0,0
	40	2	10	0,3	2	10	0,3	2	10	0,4	2	10	0,4
	60	2,8	9	0,7	2,8	9	0,7	2,8	9	0,8	2,8	10	0,9
	80	3,4	9	2,6	3,4	9	2,9	3,4	9	3,1	3,4	9	3,4
	100	4,1	9	5,4	4,1	9	6,0	4,1	9	6,6	4,1	9	7,5
3	20	5,5	9	0	5,5	9	0,0	5,6	8	0,0	5,6	8	0,0
	40	10	8	0,1	9,9	9	0,2	10,3	5	0,1	10,3	5	0,1
	60	14,5	5	0,5	14,4	6	0,8	15,1	0	0,3	15,1	0	0,4
	80	19,7	10	0,8	19,7	10	1,0	20,6	2	0,9	20,6	2	1,0

Tabela 5.4 Resultados obtidos com o SearchCol com VNS sem e com perturbação *(continuação da página anterior).*

Classe	n	Mipheur sem perturbação			Mipheur com perturbação			VNS sem perturbação			VNS com perturbação		
		z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	z	n_{opt}	t_{med}	z_{med}	n_{optMed}	t_{med}
	100	23,1	7	3,4	23,1	7	5,0	24,3	1	2,0	24,3	1	2,6
	20	1	10	0	1	10	0,0	1	10	0,0	1	10	0,0
	40	2	10	0,3	2	10	0,3	2	10	0,3	2	10	0,4
4	60	2,7	9	1,1	2,7	9	1,1	2,7	9	1,2	2,7	9	1,3
	80	3,4	9	1,3	3,4	9	1,4	3,4	9	1,6	3,4	9	1,8
	100	4	9	5,1	4	9	6,8	4	9	6,3	4	9	7,0
	20	6,7	10	0	6,7	10	0,0	7,1	6	0,0	7,08	6,2	0,0
	40	12,3	10	0,1	12,3	10	0,1	12,6	7	0,1	12,6	7	0,1
5	60	18,3	10	0,3	18,3	10	0,3	19,1	4	0,2	19,1	4	0,3
	80	25,2	8	0,5	25,1	9	0,8	26,3	1	0,5	26,3	1	0,6
	100	29,1	7	1,8	29	8	2,7	30,2	0	1,4	30,2	0	1,8
	20	1	10	0	1	10	0,0	1	10	0,0	1	10	0,0
	40	1,9	10	0,2	1,9	10	0,3	1,9	10	0,3	1,9	10	0,3
6	60	2,3	9	1,7	2,3	9	1,8	2,3	9	2,0	2,3	9	2,0
	80	3	10	2,2	3	10	2,3	3	10	2,7	3	10	2,7
	100	3,5	9	5,2	3,5	9	5,5	3,5	9	6,4	3,5	9	7,1
	20	5,7	10	0	5,7	10	0,0	6	7	0,0	5,98	7,2	0,0
	40	11,7	8	0,1	11,7	8	0,1	12	5	0,1	12	5	0,1
7	60	16,4	8	0,5	16,4	8	0,7	17	3	0,3	17	3	0,5
	80	23,3	10	0,9	23,3	10	1,3	24,3	2	0,8	24,3	2	1,1
	100	27,8	7	1,9	27,7	8	3,3	28,5	2	1,4	28,5	2	2,1
	20	6,1	10	0	6,1	10	0,0	6,4	7	0,0	6,36	7,4	0,0
	40	11,7	8	0,1	11,6	9	0,2	12,1	5	0,1	12,1	5	0,1
8	60	16,4	10	0,3	16,4	10	0,5	16,8	6	0,3	16,8	6	0,5
	80	22,6	10	0,8	22,6	10	1,4	23,6	2	0,7	23,6	2	1,0
	100	28,1	10	2,4	28,1	10	3,9	29,1	2	1,3	29,1	2	2,0
	20	14,3	10	0	14,3	10	0,0	14,5	8	0,0	14,38	9,2	0,0
	40	27,8	10	0	27,8	10	0,0	28	8	0,0	27,98	8,2	0,0
9	60	43,7	10	0	43,7	10	0,0	44,1	6	0,0	44,08	6,2	0,0
	80	57,7	10	0,1	57,7	10	0,1	58	7	0,1	58,0	7	0,1
	100	69,5	10	0,2	69,5	10	0,2	70,1	4	0,2	70,07	4,3	0,2
	20	4,6	9	0	4,6	9	0,0	4,6	9	0,0	4,6	9	0,0
	40	7,9	8	0,2	7,9	8	0,2	8,1	6	0,2	8,1	6	0,2
10	60	10,9	5	1,1	10,8	6	1,7	11,1	3	0,8	11,1	3,1	1,1
	80	13,8	4	3,8	13,8	4	6,3	14,1	2	2,5	14,1	2	3,1
	100	16,9	3	268,9	16,6	6	496,6	17,2	0	6,0	17,2	0	7,5
Soma		741,3	442	317	740,2	453	560,0	760,2	287	53,7	759,9	291,1	63,7

O SearchCol com a meta-heurística VNS com perturbações obteve melhores resultados do que o SearchCol com a meta-heurística VNS sem perturbações. Tal

como nos resultados obtidos no SearchCol com a meta-heurística PLMI sem e com perturbações, a meta-heurística VNS é bastante rápida existindo um ligeiro aumento do tempo computacional quando é utilizada a perturbação.

Comparando a utilização das meta-heurísticas Mipheur e VNS sem e com perturbações, conclui-se que Mipheur obteve melhores resultados, maior número de soluções ótimas e melhores valores médios de z. No entanto, com a meta-heurística VNS os tempos computacionais são bastante inferiores.

Apresentam-se na tabela 5.5 todos os resultados obtidos com as aplicações do SearchCol propostas para o problema 2BP. Estes resultados são comparados com os resultados computacionais de dois métodos exatos: os resultados obtidos com o modelo de afetação de Lodi et al. (2004) e os resultados obtidos com o modelo de programação inteira proposto no capítulo 3 para o problema de corte (coluna MPC). São ainda apresentados os resultados computacionais de três heurísticas e três meta-heurísticas da literatura.

As heurísticas são a *Finite First Fit* (FFF), a *Finite Best Strip* propostas por Berkey & Wang (1987) e a *Knapsack Packing* proposta por Lodi et al. (1999). As meta-heurísticas são a pesquisa tabu de Lodi et al. (1999) utilizando as heurísticas FBS e KP para pesquisar as vizinhanças e a meta-heurística *Variable Neighbourhood Descent* (VND) proposta por Alvelos et al. (2009) para diferentes variantes do problema de empacotamento bidimensional. Todas estas heurísticas e meta-heurísticas foram apresentadas no capítulo 2.

Os valores das heurísticas e meta-heurísticas da literatura foram retirados de Lodi et al. (1999) e de Alvelos et al. (2009). As coluna z/LB apresentam o valor da solução heurística/meta-heurística para cada subclasse divida pelo limite inferior proposto por Martello & Vigo (1998).

Tabela 5.5: Comparação das aplicações do SearchCol com outras heurísticas.

Classe	n	Sem perturbações			Com perturbações			Lodi et al. (2004)	MPC	FFF	FBS	KP	TS-FBS	TS-KP	VND
		Mipheur	PLMI	VNS	Mipheur	PLMI	VNS								
		z/LB	z/LB	z/LB	z/LB	z/LB		z/LB	z/LB	z/LB	z/LB	z/LB	z/LB	z/LB	z/LB
	20	1,09	1,13	1,16	1,09	1,11	1,16	1,09	1,09	1,17	1,14	1,13	1,09	1,11	1,1
	40	1,09	1,09	1,09	1,09	1,09	1,09	1,08	1,08	1,12	1,09	1,1	1,08	1,08	1,1
1	60	1,06	1,09	1,1	1,06	1,09	1,1	1,05	1,05	1,1	1,07	1,07	1,05	1,05	1,07
	80	1,03	1,07	1,07	1,03	1,06	1,07	1,03	1,03	1,08	1,06	1,06	1,04	1,04	1,04
	100	1,04	1,07	1,07	1,03	1,06	1,07	1,03	1,03	1,07	1,06	1,05	1,04	1,05	1,05
	Média	1,062	1,09	1,098	1,06	1,082	1,098	1,056	1,056	1,108	1,084	1,082	1,06	1,066	1,072
	20	1,1	1,1	1,1	1,1	1,1	1,1	1	1	1,1	1,1	1	1,1	1	1
	40	1,05	1,05	1,05	1,05	1,05	1,05	1,05	1,05	1,1	1,1	1,1	1,1	1,1	1,05
2	60	1,12	1,12	1,12	1,12	1,12	1,12	1,12	1,12	1,15	1,15	1,15	1,15	1,15	1,12
	80	1,1	1,1	1,1	1,1	1,1	1,1	1,06	1,1	1,07	1,07	1,07	1,07	1,07	1,06
	100	1,05	1,05	1,05	1,05	1,05	1,05	1,05	1,05	1,06	1,06	1,03	1,06	1,03	1,03
	Média	1,084	1,084	1,084	1,084	1,084	1,084	1,056	1,064	1,096	1,096	1,07	1,096	1,07	1,052
	20	1,2	1,21	1,22	1,2	1,21	1,22	1,17	1,17	1,2	1,18	1,18	1,18	1,18	1,2
	40	1,14	1,17	1,17	1,13	1,17	1,17	1,11	1,11	1,18	1,14	1,15	1,12	1,12	1,13
3	60	1,09	1,14	1,14	1,08	1,13	1,14	1,05	1,05	1,14	1,11	1,12	1,08	1,07	1,1
	80	1,07	1,12	1,12	1,07	1,12	1,12	1,07	1,07	1,13	1,1	1,1	1,07	1,08	1,09
	100	1,06	1,12	1,12	1,06	1,12	1,12	1,05	1,05	1,12	1,09	1,09	1,09	1,09	1,08
	Média	1,112	1,152	1,154	1,108	1,15	1,154	1,09	1,09	1,154	1,124	1,128	1,108	1,108	1,12
	20	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	40	1,05	1,05	1,05	1,05	1,05	1,05	1,05	1,16	1,1	1,1	1,1	1,1	1,1	1,1
4	60	1,17	1,17	1,17	1,17	1,17	1,17	1,13	1,79	1,2	1,2	1,2	1,2	1,2	1,2
	80	1,13	1,13	1,13	1,13	1,13	1,13	1,13	1,56	1,1	1,1	1,1	1,1	1,1	1,1
	100	1,08	1,08	1,08	1,08	1,08	1,08	1,11	1,62	1,1	1,1	1,1	1,1	1,1	1,1
	Média	1,086	1,086	1,086	1,086	1,086	1,086	1,084	1,426	1,1	1,1	1,1	1,1	1,1	1,1
	20	1,12	1,17	1,18	1,12	1,14	1,18	1,12	1,12	1,14	1,14	1,13	1,13	1,13	1,15
	40	1,09	1,12	1,12	1,09	1,11	1,12	1,09	1,09	1,11	1,11	1,09	1,09	1,09	1,12
5	60	1,06	1,11	1,11	1,06	1,1	1,11	1,06	1,06	1,11	1,1	1,1	1,06	1,07	1,09
	80	1,07	1,11	1,11	1,06	1,11	1,11	1,06	1,06	1,12	1,09	1,09	1,06	1,08	1,09
	100	1,07	1,11	1,11	1,06	1,1	1,11	1,05	1,05	1,12	1,09	1,09	1,08	1,09	1,09
	Média	1,082	1,124	1,126	1,078	1,112	1,126	1,076	1,076	1,12	1,106	1,1	1,084	1,092	1,108
	20	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	40	1,27	1,27	1,27	1,27	1,27	1,27	1,27	2	1,4	1,4	1,5	1,4	1,5	1,3
6	60	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,90	1,1	1,1	1,1	1,1	1,1	1,1
	80	1	1	1	1	1	1	1	1,67	1	1	1	1	1	1
	100	1,09	1,09	1,09	1,09	1,09	1,09	1,09	-	1,1	1,1	1,1	1,1	1,1	1,1
	Média	1,092	1,092	1,092	1,092	1,092	1,092	1,092	1,5	1,12	1,12	1,14	1,12	1,14	1,1
	20	1,08	1,13	1,13	1,08	1,1	1,13	1,08	1,08	1,1	1,1	1,1	1,08	1,08	1,08
	40	1,08	1,11	1,11	1,08	1,11	1,11	1,06	1,06	1,11	1,11	1,07	1,07	1,07	1,06
7	60	1,06	1,1	1,1	1,06	1,1	1,1	1,05	1,05	1,08	1,08	1,06	1,05	1,05	1,06
	80	1,04	1,09	1,09	1,04	1,09	1,09	1,04	1,04	1,07	1,06	1,06	1,05	1,05	1,05
	100	1,04	1,06	1,06	1,03	1,06	1,06	1,03	1,03	1,04	1,04	1,04	1,03	1,04	1,03
	Média	1,06	1,098	1,098	1,058	1,092	1,098	1,052	1,052	1,08	1,078	1,066	1,056	1,058	1,056

Tabela 5.5 Comparação das aplicações do SearchCol com outras heurísticas.

(continuação da página anterior).

Classe	n	Sem perturbações			Com perturbações			Lodi et al. (2004)	MPC	FFF	FBS	KP	TS-FBS	TS-KP	VND
		Mipheur	PLMI	VNS	Mipheur	PLMI	VNS								
		z/LB	z/LB	z/LB	z/LB	z/LB		z/LB	z/LB	z/LB	z/LB	z/LB	z/LB	z/LB	z/LB
8	20	1,11	1,16	1,16	1,11	1,15	1,16	1,11	1,11	1,17	1,16	1,12	1,12	1,12	1,13
	40	1,05	1,09	1,09	1,05	1,08	1,09	1,04	1,04	1,09	1,08	1,07	1,04	1,04	1,07
	60	1,03	1,06	1,06	1,03	1,06	1,06	1,03	1,03	1,06	1,06	1,06	1,03	1,03	1,05
	80	1,02	1,06	1,06	1,02	1,06	1,06	1,02	1,02	1,07	1,06	1,05	1,03	1,03	1,05
	100	1,03	1,07	1,07	1,03	1,07	1,07	1,03	1,03	1,06	1,06	1,04	1,04	1,04	1,05
	Média	1,048	1,088	1,088	1,048	1,084	1,088	1,046	1,046	1,09	1,084	1,068	1,052	1,052	1,07
9	20	1	1,01	1,01	1	1	1,01	1	1	1,01	1,01	1,01	1	1	1
	40	1,01	1,02	1,02	1,01	1,01	1,02	1,01	1,01	1,02	1,02	1,02	1,01	1,01	1,03
	60	1,01	1,02	1,02	1,01	1,01	1,02	1,01	1,01	1,02	1,02	1,01	1,01	1,01	1,01
	80	1,01	1,02	1,02	1,01	1,01	1,02	1,01	1,01	1,02	1,02	1,02	1,02	1,01	1,02
	100	1,01	1,02	1,02	1,01	1,01	1,02	1,01	1,01	1,02	1,01	1,01	1,01	1,01	1,01
	Média	1,008	1,018	1,018	1,008	1,008	1,018	1,008	1,008	1,018	1,016	1,014	1,01	1,008	1,014
10	20	1,15	1,15	1,15	1,15	1,15	1,15	1,13	1,13	1,14	1,14	1,16	1,14	1,14	1,13
	40	1,11	1,14	1,14	1,11	1,14	1,14	1,08	1,08	1,14	1,09	1,1	1,09	1,09	1,08
	60	1,12	1,14	1,14	1,11	1,14	1,14	1,08	1,08	1,15	1,12	1,1	1,08	1,08	1,08
	80	1,12	1,15	1,15	1,12	1,15	1,15	1,08	1,08	1,15	1,13	1,12	1,1	1,1	1,11
	100	1,1	1,12	1,12	1,08	1,12	1,12	1,07	1,07	1,14	1,1	1,08	1,08	1,07	1,07
	Média	1,12	1,14	1,14	1,114	1,14	1,14	1,088	1,088	1,144	1,116	1,112	1,098	1,096	1,094

Em Lodi et al. (1999) não são apresentados os tempos computacionais para as heurísticas FFF, FBS e KP, mas os autores referem que são sempre inferiores a 0,5 segundos. Para as meta-heurísticas pesquisa tabu, para as dez classes o tempo médio foi de 33 segundos, com o mínimo de 4 segundos e o máximo de 55 segundos. A meta-heurística VND, para as 10 classes demorou em média 0,24 segundos para obter as soluções para as 10 classes. Uma vez que foram utilizados computadores diferentes, é difícil fazer uma comparação direta entre os tempos computacionais.

As aplicações do algoritmo SearchCol com as meta-heurísticas PLMI e VNS, obtiveram soluções de qualidade inferior, mas a PLMI com perturbações obteve

16 subclasses com as mesmas soluções que o modelo de programação inteira, enquanto as aplicações do algoritmo SearchCol com PLMI sem perturbações e VNS sem e com perturbações encontraram 11 soluções de subclasses com valores iguais ao modelo de afetação. Em penúltimo e último lugar ficaram as heurísticas FBS e FFF com 5 e 4 subclasses com as mesmas soluções que o modelo inteiro, respetivamente.

As soluções obtidas pelo algoritmo SearchCol com Mipheur com perturbações foi claramente melhor que as outras heurísticas e meta-heurísticas e em 31 subclasses obteve as mesma soluções que o modelo de programação inteira, tendo ainda na classe 4 com $n = 100$ obtido melhores soluções. O algoritmo SearchCol com Mipheur sem perturbações obteve 28 subclasses com as mesmas soluções que o modelo de afetação de Lodi et al. (2004), enquanto que em terceiro lugar ficou a TS-FBS que encontrou 21 subclasses com as mesmas soluções que o modelo de afetação.

Os tempos computacionais para os resultados obtidos com o modelo de programação inteira proposto no capítulo 3 não são apresentados, mas são muito semelhantes aos tempos computacionais do modelo de afetação (tempos apresentados na tabela 5.2), à exceção das classes 4 e 6.

Nestas classes o modelo de programação inteira não encontrou a solução ótima em muitas instâncias no limite de tempo de duas horas. Estas instâncias são constituídas por itens muito pequenos tendo em conta o tamanho das placas; na classe 4 a largura e a altura dos itens pertencem ao intervalo [1 35] e as placas têm largura e altura 100; na classe 6 a largura e a altura dos itens pertencem ao intervalo [1 100] e as placas têm largura e altura 300. Para estas instâncias são geradas muitas placas residuais e muitos cortes possíveis para o modelo de programação inteira para o problema de corte, ou seja, o modelo é constiuído por muitas variáveis de decisão e muitas restrições e torna-se de difícil resolução para

o Cplex 12.1.

5.7 Conclusões

Neste capítulo foi aplicado um algoritmo SearchCol ao problema de empacotamento bidimensional.

O SearchCol combina geração de colunas com meta-heurísticas, sendo um método geral para resolver problemas de otimização combinatória decomponíveis. Assume que a estrutura da solução para um problema resulta da combinação de soluções de (sub)problemas mais pequenos que são gerados pela geração de colunas. Uma pesquisa meta-heurística é realizada no espaço definido pelas colunas geradas pelos subproblemas da geração da colunas. Cada iteração do Searchcol é constituída por três fases: geração de colunas (perturbada), pesquisa meta-heurística e perturbação.

Neste capítulo foi proposto um novo modelo de decomposição e algoritmo de geração de colunas para o problema de empacotamento bidimensional. No modelo de decomposição proposto, cada subproblema está associado a um item, existindo deste modo tantos subproblemas quantos os itens. Uma solução de um subproblema k é um conjunto de itens posicionados numa placa iniciada pelo item k.

Para a pesquisa meta-heurística foram utilizadas a pesquisa local multi-início, a pesquisa em vizinhanças variáveis e a pesquisa Mipheur; foram também utilizados dois tipos de perturbações, uma para quando o valor da solução incumbente foi atualizada na solução anterior, e outra para quando solução incumbente não foi melhorada.

Os resultados computacionais são bastante interessantes para a aplicação do SearchCol, em particular quando na fase de pesquisa é resolvido o problema

inteiro de forma exata. Para o algoritmo SearchCol com Mipheur das 500 soluções obtidas 453 eram ótimas, sendo além disso os tempos computacionais bastante inferiores aos obtidos pelo modelo de programação inteira da literatura.

O modelo de decomposição é bastante interessante, a comprovar isso estão os tempos computacionais despendidos com a geração de colunas, que foram pequenos.

Para as aplicações do SearchCol em que a fase de pesquisa foi otimizada pelas meta-heurísticas pesquisa multi-início e pesquisa em vizinhanças variáveis o número de soluções ótimas obtidas foi menor, apesar destes procedimentos serem muito rápidos, podendo ser bastante úteis quando se pretende obter uma solução rapidamente, à custa da degradação da sua qualidade.

Capítulo 6

Conclusões

Conclusões

Nesta tese, foram abordadas diferentes variantes dos problemas de corte e empacotamento a duas dimensões. Foram propostos novos modelos de programação inteira e a aplicação do algoritmo SearchCol ao problema de empacotamento. Foram realizados testes computacionais exaustivos com instâncias da literatura e instâncias reais, de modo a provar a eficiência e a eficácia dos modelos e métodos propostos. Neste último capítulo apresentam-se as conclusões finais do trabalho desenvolvido.

No problema de corte bidimensional abordado na capítulo 3 pretende-se cortar um conjunto de itens retangulares a partir do menor número possível de placas também retangulares, permitindo que os padrões de corte sejam de um de quatro tipos (dois-estágios exato, dois-estágios não-exato, três-estágios exato e três-estágios não-exato). Para estes problemas, as abordagens mais frequentes são: geração de colunas, modelos polinomiais de programação inteira e heurísticas.

Para este problema propusemos uma abordagem diferente, um modelo de programação inteira com um número pseudo-polinomial de variáveis e restrições.

Para comprovar a eficiência do modelo proposto foi realizado um vasto conjunto de testes computacionais com instâncias reais e adaptadas da literatura para quatro variantes do problema. Foram realizadas em paralelo comparações com modelos de geração de colunas e modelos de programação inteira de tamanho polinomial, concluindo-se que o modelo proposto foi capaz de resolver um grande número de instâncias em tempos computacionais aceitáveis, superando assim todos os outros métodos de otimização testados.

Além da eficiência apresentada, o modelo tem ainda outro aspeto positivo, a flexibilidade que apresenta para incorporar características que podem ser de elevada importância em termos práticos. Deste modo, foi apresentado um conjunto

de extensões para o modelo, em particular: a consideração da existência de placas de vários tamanhos, a incorporação de valores para as placas remanescentes, a inclusão dos comprimentos dos cortes e a rotação dos itens.

Tipicamente no mundo industrial, as placas residuais que sobram do corte dos itens das placas grandes, são consideradas desperdício, o que justifica o facto de na literatura o critério de otimização mais utilizado ser a minimização dos desperdícios.

No entanto, parece importante em termos económicos guardar as placas residuais em inventário, desde que tenham um tamanho aceitável, e reutilizá-las num período de planeamento futuro para cortar outros itens.

Se ao armazenamento de placas residuais acrescentarmos a possibilidade de antecipar a produção de itens e proceder ao seu armazenamento para satisfazer a procura em períodos subsequentes do horizonte de planeamento, passamos para um problema que visa a minimização dos custos num sentido global de todo o processo de produção.

É neste contexto que surge o problema integrado de corte bidimensional e dimensionamento de lotes apresentado no capítulo 4. Para este problema foram propostos dois modelos de programação inteira onde a função objetivo reflete a minimização dos custos de cortar os itens, os custos das placas utilizadas e também os custos de armazenar itens e placas residuais (sobras).

Os modelos propostos foram testados com um conjunto de instâncias reais e comparados com duas heurísticas, uma consiste na antecipação da produção de todos os itens para o primeiro período, a segundo consiste em resolver em cada período do horizonte de planeamento um problema de corte a duas dimensões isoladamente.

Foram considerados 4 horizontes de planeamento diferentes, e as instâncias

reais testadas foram resolvidas otimamente em tempo computacional aceitável. Comparando as soluções ótimas com as soluções heurísticas, observa-se que quando o número de períodos aumenta, os ganhos obtidos pelas soluções ótimas também aumentam.

Os modelos de programação inteira propostos foram desenvolvidos para uma variante do problema *2CS-LSP* na qual é permitida a antecipação da produção dos itens e o armazenamento de sobras, com o objetivo de minimizar os custos de produção e de armazenamento. Na literatura esta característica nunca foi abordada, contudo existem alguns problemas similares. Além de ter sido apresentada uma abordagem que pode permitir obter melhores resultados em termos de utilização da matéria-prima, é importante salientar que foram obtidas soluções exatas para o modelo proposto, enquanto nos trabalhos anteriores foram apenas obtidos resultados heurísticos.

O problema de empacotamento bidimensional abordado na tese consiste em posicionar um conjunto de itens retangulares no menor número possível de placas também retangulares e do mesmo tamanho. Os itens têm ainda de ser empacotados em níveis.

Existe uma forte semelhança entre este problema e o problema de corte de dois-estágios não-exato, a diferença reside nas características dos itens a empacotar que em oposição ao problema de corte têm quase todos os tamanhos diferentes e são considerados individualmente, logo a procura é igual a um. Esta diferença faz com que existam muitas mais combinações geométricas dos itens nas placas.

Foi proposto um novo modelo de decomposição para o problema de empacotamento bidimensional em que cada subproblema propõe padrões de empacotamento inicializados por um item diferente e os subproblemas são resolvidos através de uma adaptação da abordagem clássica de Gilmore & Gomory (1965) para o

problema de corte bidimensional com padrões do tipo dois-estágios não-exato.

Para este problema foi proposta a aplicação do algoritmo SearchCol que combina geração de colunas com meta-heurísticas. O SearchCol considera que uma solução para um problema resulta da combinação de soluções de (sub)problemas mais pequenos que são gerados pela geração de colunas. Com as colunas obtidas durante a geração de colunas define-se um espaço de pesquisa e são utilizadas meta-heurísticas nesse espaço. Cada iteração do Searchcol é constituída por três fases: geração de colunas (perturbada), pesquisa meta-heurística e perturbação.

Para a pesquisa meta-heurística foram utilizadas a pesquisa local multi-início, a pesquisa em vizinhanças variáveis e a heurística MipH, foram também utilizados dois tipos de perturbações, uma para quando o valor da solução incumbente foi atualizada na solução anterior e outra para quando solução incumbente não foi melhorada.

Foram realizados testes computacionais num conjunto de 500 instâncias da literatura, os resultados obtidos foram bastante interessantes, dado que foram obtidas 453 soluções ótimas num tempo médio de 56 segundos quando na fase de pesquisa é resolvido o problema inteiro de forma exata. Além disso, o modelo de decomposição mostrou-se bastante interessante uma vez que os tempos computacionais gastos na geração de colunas foram desprezáveis.

Para as aplicações do SearchCol em que a fase de pesquisa foi otimizada pelas meta-heurísticas pesquisa multi-início e pesquisa em vizinhanças variáveis o número de soluções ótimas obtidas foi menor, no entanto estes procedimentos foram muito rápidos. Podendo ser úteis quando se pretende uma solução rapidamente, á custa da degradação da sua qualidade.

Bibliografia

Alvarez-Valdes, R., M, R., Parajón, A. & Tamarit, J. M. (2002), 'A computational study of LP-based heuristics algorithms for the two-dimensional guillotine cutting stock problems', *OR Spectrum* 24.

Alvarez-Valdes, R., M, R., Parajón, A. & Tamarit, J. M. (2007), 'Grasp and path relinking for the two-dimensional two-staged cutting stock problem', *Journal on Computing* 19(2), 261–272.

Alvelos, F., Chan, T. M., Vilaça, P., Gomes, T., Silva, E. & Carvalho, J. M. V. d. (2009), 'Sequence based heuristics for two-dimensional bin packing problems', *Engineering Optimization* 41(8), 773–791.

Alvelos, F., de Sousa, A. & Santos, D. (2010), Searchcol: Metaheuristic search by column generation, *in* M. Blesa, C. Blum, G. Raidl, A. Roli & M. Sampels, eds, 'Hybrid Metaheuristics', Vol. 6373 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 190–205.

Arbib, C. & Marinelli, F. (2005), 'Integrating process optimization and inventory planning in cutting-stock with skiving option: An optimization model and its application', *European Journal of Operational Research* 163, 617–630.

Arenales, M. & Morabito, R. (1995), 'An AND/OR- graph approach to the solution of two-dimensional non-guillotine cutting problems', *European Journal of Operational Research* 84, 599–617.

Beasley, J. E. (1985), 'Algorithms for unconstrained two-dimensional guillotine cuttting', *Journal of the Operational Research Society* 36, 297–306.

Belov, G. & Scheithaeur, G. (2006), 'A branch-and-cut-and-price algorithm for one-dimensional stock cutting and two-dimensional two-stage cutting', *European Journal of Operational Research* 171, 85–106.

Berkey, J. O. & Wang, P. Y. (1987), 'Two-dimensional finite bin-packing algorithms', *The Journal of the Operational Research Society* 38(5), 423–429.

Boschetti, M. A. & Mingozzi, A. (2003*a*), 'The two-dimensional finite bin packing problem, part I: New lower bounds for the oriented case', *4OR* 1, 27–42.

Boschetti, M. A. & Mingozzi, A. (2003*b*), 'The two-dimensional finite bin packing problem. part II: New lower and upper bounds', *4OR* 1, 135 – 147.

Caprara, A., Fischetti, M. & Toth, P. (1999), 'A heuristic method for the set covering problem', *Operations Research* 47(5), 730–743.

Chan, T. M., Alvelos, F., Silva, E. & Carvalho, J. M. V. d. (2009), A combined local search approach for the two-dimensional bin packing problem, *in* 'Proceedings of the EU/MEeting 2009 European Chapter on Metaheuristics Workshop', pp. 153–158.

Chan, T. M., Alvelos, F., Silva, E. & Carvalho, J. M. V. d. (2011*a*), Heuristics for two-dimensional bin-packing problems, *in* J. D. Irwin & B. M. Wilamowski, eds, 'The Industrial Electronics Handbook', Vol. 5 of *Intelligent Systems*, CRC and IEEE Press.

Chan, T. M., Alvelos, F., Silva, E. & Carvalho, J. M. V. d. (2011*b*), 'Heuristics with stochastic neighborhood structures for 2-dimensional bin-packing problems', *Asia-Pacific Journal of Operations Research* 28(2), 255–278.

Charalambous, C. & Fleszar, K. (2011), 'A constructive bin-oriented heuristic for

the two-dimensional bin packing problem with guillotine cuts', *Computers and Operations Research* 38.

Chen, Y. (2008), 'A recursive algorithm for constrained two-dimensional cutting problems', *Computational Optimization and Applications* 41, 337–348.

Christofides, N. & Hadjiconstantinou, E. (1995), 'An exact algorithm for orthogonal 2-d cutting problems using guillotine cuts', *European Journal of Operational Research* 83, 21–38.

Christofides, N. & Whitlock, C. (1977), 'An algorithm for two-dimensional cutting problems', *Operations Research* 25(1), 30–44.

Chung, F. R. K., Garey, M. R. & Johnson, D. S. (1982), 'On packing two-dimensional bins', *Journal on Algebraic and Discrete Methods* 3(1).

Cintra, G. F., Miyazawa, F. K., Wakabayashi, Y. & Xavier, E. C. (2008), 'Algorithms for two-dimensional cutting stock and strip packing problems using dyanmic programming and column generation', *European Journal of Operational Research* 191, 59–83.

Cui, Y. (2008), 'Heuristic and exact algorithms for generating homogenous constrained three-staged cutting patterns', *Computers & Operations Research* 35, 212–225.

Dantzig, G. B. & Wolfe, p. (1960), 'Decomposition principle for linear programs', *Operations Research* 8, 101.

Dyckhoff, H. (1981), 'A new linear programming approach to the cutting stock problem', *Operations Research* 29(6), 1092–1104.

Dyckhoff, H. (1990), 'A typology of cutting and packing problems', *European Journal of Operational Research* 44, 145–159.

Erjavec, J., Gradisar, M. & Trkman, P. (2009), 'Renovation of the cutting process', *International Journal of Production Research* 47(14), 3979–3996.

Farley, A. (1988), 'Mathematical programming models for cutting-stock problems in the clothing industry', *The Journal of the Operational Research Society* 39(1), 41–53.

Ford, L. R. & Fulkerson, D. R. (1958), 'Constructing maximal dynamic flows from static flows', *Operations Research* 6(3), pp. 419–433.

Garey, M. R. & Johnson, D. S. (1990), *Computers and Intractability; A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., New York, NY, USA.

Gilmore, P. C. & Gomory, R. E. (1961), 'A linear programming approach to the cutting-stock problem', *Operations Research* 9(6), 849–859.

Gilmore, P. C. & Gomory, R. E. (1963), 'A linear programming approach to the cutting stock problem - part II', *Operations Research* 11(6), 863–888.

Gilmore, P. C. & Gomory, R. E. (1965), 'Multistage cutting stock problems of two and more dimensions', *Operations Research* 13(1), 94–120.

Gilmore, P. C. & Gomory, R. E. (1967), 'The theory and computation of knapsack functions', *Operations Research* pp. 1045–1074.

Gramani, M. C., França, P. M. & Arenales, M. N. (2010), 'A linear optimization approach to the combined production planning model', *Journal of the Franklin Institute* doi:10.1016/j.franklin.2010.05.010.

Gramani, M. C. N. & França, P. M. (2006), 'The combined cutting stock and lot-sizing problem in industrial processes', *European Journal of Operational Research* 174, 509–521.

Gramani, M. C. N., França, P. M. & Arenales, M. N. (2009), 'A lagrangian relaxation approach to a coupled lot-sizing and cutting stock problem', *International Journal of Production Economics* 119, 219–227.

Haessler, R. W. (1971), 'A heuristic programming solution to a nonlinear cutting stock problem', *Management Science* 17(12), 793–802.

Hendry, L. C., Fok, K. K. & Shek, K. W. (1996), 'A cutting stock and scheduling problem in the copper industry', *The Journal of the Operational Research Society* 47(1), 38–47.

Herz, J. C. (1972), 'A recursive computational procedure for two-dimensional stock-cutting', *IBM Journal of Research and Development* pp. 462–469.

Hifi, M. (1997), 'The DH/KD algorithm: a hybrid approach for unconstrained two-dimensional cutting problems', *European Journal of Operational Research* 97, 41–52.

Hifi, M. (2001), 'Exact algorithms for large-scale unconstrained two and three stage cutting problems', *Computational Optimization and Applications* 18, 63–88.

Hifi, M. (2004), 'Dynamic programming and hill-climbing', *Journal of Combinatorial Optimization* 8, 65–84.

Kantorovich, L. V. (1960), 'Mathematical methods of organizing and planning production', *Management Science* 6, 366–422.

Lodi, A., Martello, S. & Monaci, M. (2002), 'Two-dimensional packing problems: A survey', *European Journal of Operational Research* 141, 241–252.

Lodi, A., Martello, S. & Vigo, D. (1998), Neighborhood search algorithm for the guillotine non-oriented two-dimensional bin packing problem, *in* 'Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization', Kluwer Academic Publishers, pp. 125–139.

Lodi, A., Martello, S. & Vigo, D. (1999), 'Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems', *Journal on Computing* 11, 345–357.

Lodi, A., Martello, S. & Vigo, D. (2002), 'Recent advances on two-dimensional bin packing problems', *Discrete Applied Mathematics* 123, 379–396.

Lodi, A., Martello, S. & Vigo, D. (2004), 'Models and bounds for two-dimensional level packing problems', *Journal of Combinatorial Optimization* 8, 363–379.

Lodi, A. & Monaci, M. (2003), 'Integer linear programming models for 2-staged two-dimensional knapsack problems', *Mathematical Programming* Ser B 94, 257–278.

Martello, S. & Vigo, D. (1998), 'Exact solution of the two-dimensional finite bin packing problem', *Management Science* 44(3), 388–392.

Menon, S. & Schrage, L. (2002), 'Order allocation for stock cutting in the paper industry', *Operations Research* 50(2), 324–332.

Mladenovic, N. & Hansen, P. (1997), 'Variable neighborhood search', *Computers & Operations Research* 24(11), 1097–1100.

Monaci, M. & Toth, P. (2006), 'A set-covering-based heuristic approach for bin-packing problems', *Journal on Computing* 18, 71–85.

Morabito, R. & Arenales, M. (1994), 'An AND/OR-graph approach to the container loading problem', *International Transactions in Operational Research* 1(59-73).

Morabito, R. & Arenales, M. N. (1996), 'Staged and constrained two-dimensional guillotine cutting problems: An AND/OR - graph approach', *European Journal of Operational Research* 94, 548–560.

Morabito, R. & Pureza, V. (2010), 'A heuristic approach based on dynamic programming and AND/OR-graph search for the constrained two-dimensional guillotine cutting problem', *Annals of Operations Research* 179, 297–315.

Nonas, S. L. & Thorstenson, A. (2000), 'A combined cutting-stock and lot-sizing problem', *European Journal of Operational Research* 120, 327–342.

Nonas, S. L. & Thorstenson, A. (2008), 'Solving a combined cutting-stock and lot-sizing problem with a column generating procedure', *Computers & Operations Research* 35, 3371–3392.

Oliveira, J. F. & Ferreira, J. S. (1990), 'An improved version of Wang's algorithm for two-dimensional cutting problems', *European Journal of Operational Research* 44, 256–266.

Oliveira, J. F. & Ferreira, J. S. (1994), 'A faster variant of the gilmore and gomory technique for cutting stock problems', *JORBEL - Belgium Journal of Operations Research, Statistics and Computer Science* 34 (1), 23–38.

Parada, V., Alvargenga, A. G. d. & Diego, J. d. (1995), 'Exact solutions for constrained two-dimensional cutting problems', *European Journal of Operational Research* 84(633-644).

Parada, V., Muñoz, R. & Gómes, A. (1995), An hybrid genetic algortihm for the two-dimensional cutting problemin evolutionary algorithms in management applications, Springer, Berlin.

Parada, V., Palma, R., Sales, D. & Gómes, A. (2000), 'A comparative numerical analysis for the guillotine two-dimensional cutting problem', *Annals of Operations Research* 96, 245–254.

Parada, V., Sepúlveda, M., Solar, M. & Gómes, A. (1998), 'Solution for the constrained guillotine cutting problem by simulated annealing', *Computers & Operations Research* 25, 37–47.

Pinto, S. V., Alvelos, F., Silva, E. & Carvalho, J. M. V. d. (2009), 'Heurísticas para empacotamento em placas de vários tamanhos', *IO2009 -14.º Congresso da Associação Portuguesa de Investigação Operacional - Vencer novos desafios nos transportes e mobilidade!* pp. 171–178.

Pisinger, D. & Sigurd, M. (2007), 'Using decomposition techniques and constraint programming for solving the two-dimensional bin-packing problem', *Journal on Computting* 19(1), 36–51.

Poldi, K. C. (2007), O problema de corte de estoque multiperíodo, Phd thesis, Universidade de São Paulo.

Poltroniere, S. C., Poldi, K. C., Toledo, F. M. B. & Arenales, M. N. (2008), 'A coupling cutting stock-lot sizing problem in the paper industry', *Annals of Operations Research* 157, 91–104.

Puchinger, J. & Raidl, G. R. (2007), 'Models and algorithms for three-stage two-dimensional bin packing', *European Journal of Operational Research* 183, 1304–1327.

Riheme, J., Scheithaeur, G. & Terno, J. (1996), 'The solution of two-stage guillotine cutting stock problems having extremely varying order demands', *European Journal of Operational Research* 91, 543–552.

Silva, E., Alvelos, F. & Carvalho, J. M. V. d. (2010), 'A new integer programming for two- and three-stage two dimensional cutting stock problems', *European Journal of Operational Research* 205, 699–708.

Silva, E., Alvelos, F. & Carvalho, J. M. V. d. (2011*a*), 'Integrating two-dimensional cutting stock and lot-sizing problems', *Submetido* .

Silva, E., Alvelos, F. & Carvalho, J. M. V. d. (2011*b*), 'Problema integrado de corte bidimensional e dimensionamento de lotes', *Livro de Actas do 15º Congresso da Associação Portuguesa de Investigação Operacional IO 2011* pp. 91–102.

Valério de Carvalho, J. (2005), 'Using extra dual cuts to accelerate column generation', *Journal on Computing* 17, 175–182.

Vanderbeck, F. (2001), 'A nested decomposition approach to a three-stage, two-dimensional cutting-stock problem', *Management Science* 47(6), 864–879.

Viswanathan, K. V. & Bagchi, A. (1993), 'Best-first searcch methods for constrained two-dimensional cutting stock problems', *Operations Research* 41(4), 768–776.

Wang, P. Y. (1983), 'Two algorithms for constrained two-dimensional cutting stock problems', *Operations Research* 31(3), 573–586.

Wäscher, G., Haubner, H. & Schumann, H. (2007), 'An improved typology of cutting and packing problems', *European Journal of Operational Research* 183, 1109–1130.

Anexos

Anexo I

Para a afinação dos parâmetros utilizados para os algoritmos SearchCol aplicados ao problema de empacotamento bidimensional, apresentados no capítulo 5, foi realizado um conjunto de testes computacionais preliminares.

Para estes testes foram utilizadas as classes 1 e 10 da literatura apresentadas no referido capítulo.

Os diferentes parâmetros são apresentados para a meta-heurística pesquisa local multi-início, pesquisa em vizinhanças variáveis e para a heurística Mipheur com perturbações.

Pesquisa local multi-início

Apresenta-se em seguida a lista de parâmetros utilizada para a pesquisa multi-início, assim como as diferentes alternativas que podem ser utilizadas.

- **Alpha.**

  Controla a *greediness* na construção aleatória, igual a 1 é puramente aleatório e 0 é *greedy*, pode variar no intervalo [0 1].

- **Geração da primeira solução inicial.**

  0) A solução é construída através do arredondamento da solução da relaxação linear. É escolhida a solução com maior valor.

  1) A solução é construída selecionando aleatoriamente uma solução para cada subproblema, cada uma com igual probabilidade de ser selecionada.

  2) A solução é construída selecionando aleatoriamente uma solução para cada subproblema; a probabilidade de uma solução de um subproblema ser selecionada é igual ao seu valor na solução da relaxação linear.

3) A solução é construída selecionando a última solução guardada para cada subproblema.

4) A solução é construída selecionando a primeira solução obtida por cada subproblema.

5) A solução é construída selecionando a solução obtida da última vez que cada subproblema foi resolvido

6) A solução é construída de uma forma gulosa.

7) A solução é construída de uma forma gulosa aleatória.

– **Avaliação de soluções admissíveis.**

2) As soluções admissíveis são comparadas tendo por base os custos originais.

3) As soluções admissíveis são comparadas tendo por base os custos reduzidos.

– **Avaliação de soluções não-admissíveis.**

1) As soluções não-admissíveis são comparadas tendo por base o valor artificial no problema mestre.

2) As soluções não-admissíveis são comparadas tendo por base o número de restrições violadas.

3) As soluções não-admissíveis são comparadas tendo por base a soma do valor das restrições violadas (valor total da violação).

4) As soluções não-admissíveis são comparadas tendo por base o número de violações vezes 1000 mais o valor total da violação.

– **Estratégias de pesquisa local.**

0) Primeira melhoria.

1) Melhor descida.

– **Estrutura de vizinhança.**

1) Estrutura de vizinhança com uma modificação.

2) Estrutura de vizinhança com no máximo duas modificações.

– **Número máximo de iterações sem melhoria.**

– **Geração da solução inicial quando a solução incumbente não foi atualizada.**

1) A solução é construída selecionando aleatoriamente uma solução para cada subproblema, cada um com igual probabilidade de ser selecionado.

2) A solução é construída selecionando aleatoriamente uma solução para cada subproblema, a probabilidade de uma solução de um subproblema ser selecionada é igual ao seu valor na solução da relaxação linear.

7) A solução é construída de uma forma gulosa aleatória.

8) A solução é construída fazendo uma alteração na solução incumbente.

9) A solução é construída fazendo uma alteração na solução ótima local.

11) A solução é construída fazendo uma alteração na solução incumbente tendo por base a frequência.

– **Geração da solução inicial quando a solução incumbente foi atualizada.**

1) A solução é construída selecionando aleatoriamente uma solução para cada subproblema, cada um com igual probabilidade de ser selecionado.

2) A solução é construída selecionando aleatoriamente uma solução para cada subproblema, a probabilidade de uma solução de um subproblema ser selecionada é igual ao seu valor na solução da relaxação linear.

7) A solução é construída de uma forma gulosa aleatória.

8) A solução é construída fazendo uma alteração na solução incumbente.

10) A solução é construída fazendo uma alteração na solução incumbente tendo por base a memória de curto prazo.

– **Intensidade das alterações.**

Quando uma solução é alterada de modo a construir uma nova solução, este parâmetro indica a intensidade desta alteração, utilizando a proporção de soluções de subproblemas a modificar, igual a 1 a solução é completamente aleatória, igual a 0 a solução anterior não sofre qualquer alteração, pode variar no intervalo [0 1].

– ***Path relinking.***

0) Se o *path relinking* não é utilizada.

1) Se o *path relinking* é utilizada.

Na tabela seguinte apresentam-se os parâmetros utilizados na primeira iteração de afinação de parâmetros e os parâmetros finais.

Os testes foram realizados testando cada um dos parâmetros e fixando os que encontravam melhores resultados. Os resultados obtidos, z soma dos valores médias de cada classe e n_{opt} número de soluções ótimas, foram comparados com o as soluções obtidas com o modelo de Lodi et al. (2004). Como apresentado na tabela 6, começou-se por testar as diferentes alternativas para o parâmetro geração da primeira solução inicial, os melhores resultados foram obtidos com a

Tabela 1: Parâmetros da pesquisa multi-início.

Parâmetros	**Valores**	
	1.ª iteração	Finais
alpha	0,2	0,2
Geração da primeira solução inicial	0	4
Avaliação de soluções admissíveis	2	2
Avaliação de soluções não-admissíveis	2	2
Tipos de estratégia de pesquisa local	0	1
Estrutura de vizinhança utilizada na pesquisa local	1	1
Número máximo de iterações sem melhoria	0	1
Geração da solução inicial quando a solução incumbente não foi atualizada	2	2
Geração da solução inicial quando a solução incumbente foi atualizada	2	1
Intensidade das alterações	0,01	0,01
Religamento de caminhos	0	0

heurística 4, na qual a solução é construída selecionando a primeira solução de cada subproblema. Procedeu-se de igual modo para os restantes parâmetros, na tabela as linhas a cinzento significam que existiu uma melhoria do valor médio das soluções, do número de ótimos obtidos ou do tempo, para o parâmetro em análise nessa linha.

Afinação de parâmetros para a heurística PLMI

Parâmetros		PLMI		
		z	n_{opt}	t
Geração da primeira solução inicial	0	16,51	32	80,863
	1	23	2	94,045
	2	17,05	27	80,469
	3	25,12	0	95,631
	4	16,03	38	99,134
	5	20,21	5	82,393
	6	18,54	11	84,707
	7	19,63	4	86,442
Avaliação de soluções não-admissíveis	4	16,03	38	100,024

Parâmetros		PLMI		
		z	n_{opt}	t
Estratégia de pesquisa local	1	16,03	38	92,174
Estrutura de vizinhança	2	16,03	38	1585,804
Número máximo de iterações sem melhoria	5	16,02	39	97,661
	1	16,02	39	93,706
Solução inicial quando a incumbente atualizada	1	16,03	38	103,781
	7	16,03	38	99,785
	8	16,03	38	92,737
	10	16,03	38	93,158
Solução inicial quando a incumbente não atualizada	1	16,01	39	93,378
	7	16,03	38	93,467
	8	16,02	39	93,376
	9	16,03	38	93,405
	11	16,02	39	94,181
Intensidade das alterações	0,05	16,03	38	94,006
	0,02	16,02	39	94,004
Religamento de caminhos	1	16,03	38	94,014

Pesquisa em vizinhanças variáveis

Para a heurística pesquisa em vizinhanças variáveis os parâmetros possíveis são apresentados em seguida.

- **Estrutura de vizinhança.**
- **Estratégias de pesquisa local.**
- **Geração da primeira solução inicial.**
- **Alpha**
- **Número máximo de estruturas de vizinhança**

Estes parâmetros podem ser testados com as diferentes alternativas apresentadas para a heurística pesquisa local multi-início. Procedemos do mesmo modo que para a meta-heurística pesquisa local multi-início. Os resultados podem ser consultados nas tabelas 3 e 4.

Tabela 3: Parâmetros para a pesquisa VNS.

Parâmetros	**Valores**	
	1.ª iteração	Finais
Estrutura de vizinhança	1	1
Estratégias de pesquisa local	1	1
Geração da solução inicial	0	4
Alpha	0,2	0,2
Número máximo de estruturas de vizinhança	5	3

Para os parâmetros estrutura de vizinhança, estratégias de pesquisa local e alpha foram adotados da pesquisa local multi-início após a afinação de parâmetros. Para os restantes, geração da solução inicial e número máximo de estruturas

de vizinhança, os resultados para as diferentes alternativas são apresentados na tabela 4.

Afinação de parâmetros para a heurística VNS

Parâmetros		VNS		
		z	n_{opt}	t
Estrutura de vizinhança	0	18,16	22	89,737
	1	22,03	4	95,188
	2	18,53	12	83,001
	3	23,17	1	95,269
	4	16,03	38	95,826
	5	19,64	7	84,750
	6	18,31	15	87,913
	7	19,16	5	89,080
Número máximo de estruturas de vizinhança	3	16,03	38	90,568
	10	16,03	38	122,645

Perturbações

Em seguida são apresentados os parâmetros utilizados para perturbar o problema mestre.

- **Número máximo de iterações sem melhoria da solução corrente.**
- **Perturbação para solução não-admissível.**

  0) Torna a solução admissível tendo por base a solução incumbente.

  1) Torna a solução admissível tendo por base toda a informação
- **Perturbação quando a solução incumbente foi atualizada na iteração anterior.**

  1) Perturbação baseada na informação dual.

  2) Perturbação baseada na geração de colunas com 1s.

  3) Perturbação baseada na geração de colunas com 1s e zeros.

  4) Perturbação baseada na solução incumbente.

  5) Perturbação baseada na diferença.

  6) Perturbação baseada na solução recente.
- **Proporção de 1s.**

  Perturbação definida no intervalo [0 1].
- **Perturbação quando a solução incumbente não foi atualizada na iteração anterior.**

  1) Perturbação baseada na informação dual.

  2) Perturbação baseada na geração de colunas com 1s.

3) Perturbação baseada na geração de colunas com 1s e zeros.

4) Perturbação baseada na solução incumbente.

6) Perturbação baseada na solução recente.

7) Perturbação baseada na frequência.

As perturbações são utilizadas na terceira fase do algoritmo SearchCol, o que implica que já tenha sido resolvida uma geração de colunas (perturbada ou não) e tenha sido utilizada também uma heurística para obter uma solução inteira. Deste modo, a afinação dos parâmetros das perturbações é efetuada com a utilização da heurística Mipheur.

Tal como nas afinações de parâmetros anteriores são apresentadas duas tabelas, uma com os parâmetros iniciais e finais e a outra com os valores obtidos em cada teste de parâmetros.

Tabela 5: Parâmetros da perturbações

Parâmetros	**Valores**	
	1.ª iteração	Finais
Número máximo de iterações sem melhoria da incumbente	3	1
Perturbação para solução não-admissível	1	1
Perturbação quando incumbente atualizada na iteração anterior	2	4
Proporção de 1s	0,1	0,1
Perturbação quando incumbente não atualizada na iteração anterior	2	5

Afinação dos parâmetros para as perturbações

Parâmetros		Mipheur		
		z	n_{opt}	t
Número máximo de iterações sem melhoria da incumbente	3	15,48	9128,65	74^a
	1	15,58	5565,249	77
	5	15,27	14112,782	74^b

Parâmetros		Mipheur		
		z	n_{opt}	t
Perturbação para solução não-admissível	0	15,58	5605,735	77
Perturbação quando incumbente atualizada na iteração anterior	1	15,58	5237,853	77
	3	15,57	9035,07	78
	4	15,54	9662,046	81
	5	15,59	5503,223	76
	6	15,61	5982,604	74
Proporção de 1s	0,05	15,54	17577,241	81
	0,3	15,57	15526,162	78
Perturbação quando incumbente não atualizada na iteração anterior	1	15,56	9056,76	79
	3	15,56	8576,676	79
	4	15,54	7520,421	81
	5	15,54	6810,701	81
	7	15,54	7457,314	81

[a] - 8 instâncias não foram resolvidas

[b] - 9 instâncias não foram resolvidas