

JOSÉ FERNANDO DA COSTA OLIVEIRA

**OPTIMIZAÇÃO
EM CORTES
RECTANGULARES**

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
Departamento de Engenharia Electrotécnica e de Computadores

JANEIRO 1990

Tese submetida para satisfação parcial
dos requisitos do programa de mestrado
em
Engenharia Electrotécnica e de Computadores
(Sistemas)

Tese realizada sob a supervisão do

Doutor Eng^o José António Soeiro Ferreira

Professor Associado do Departamento de Engenharia Electrotécnica e de Computadores da

Faculdade de Engenharia da Universidade do Porto

À São

Agradecimentos

Desejo expressar os meus agradecimentos ao meu orientador, Professor Doutor José António Soeiro Ferreira, pelo apoio e incentivo, científico e humano, dados durante a realização do trabalho que originou esta tese, assim como na fase de escrita e revisão da dissertação propriamente dita.

Gostaria ainda de agradecer aos meus colegas Maria Antónia Carravilla, Maria Margarida Ferreira, Maria do Rosário Pinho e José Carlos Alves pelo apoio, compreensão, espírito de entreajuda e paciência que constantemente manifestaram.

Índice

	Pág.
1- Introdução	1
1.1- Considerações gerais	1
1.2- Âmbito da dissertação	2
1.3- Estrutura da dissertação	3
PARTE I - O PROBLEMA DOS CORTES RECTANGULARES	
2- O Problema dos Cortes: estrutura e relação com problemas afins	7
3- Classificação dos Problemas de Cortes	10
4- Abordagens à resolução do Problema dos Cortes rectangulares	23
4.1- Classificação das técnicas	23
4.2 - Levantamento bibliográfico	26
PARTE II - TÉCNICAS UNI-PADRÃO	
5 - Algoritmos uni-padrão para uma abordagem optimizante do Problema dos Cortes rectangulares	49
5.1- Formulação e modelos	50
5.2- O algoritmo de Christofides e Whitlock	52

5.3- O algoritmo de Wang	60
5.4- O algoritmo de Wang modificado	67
6- Comparação dos algoritmos: resultados computacionais	70
 PARTE III - TÉCNICAS MULTI-PADRÃO	
7- A aplicação da Programação Linear com geração atrasada de colunas ao Problema dos Cortes	89
8- Uma abordagem diferente à técnica da geração de colunas: a geração rápida de colunas ...	95
9- Estudo comparativo do desempenho da geração atrasada de colunas e da geração rápida de colunas: resultados computacionais	105
10- Conclusões	117
 Apêndice A- O algoritmo de pesquisa lexicográfica de Gilmore e Gomory para Problemas de Cortes a uma dimensão	
	121
 Apêndice B- O algoritmo de Programação Dinâmica de Gilmore e Gomory para Problemas de Cortes rectangulares, com padrões em n fases	
	125
 Apêndice C- A heurística FFDH: "First Fit Decreasing Height"	
	130
 Apêndice D- Resultados detalhados dos testes computacionais efectuados no capítulo 9	
	132
Bibliografia	140

Capítulo 1

Introdução

1.1 - Considerações gerais.

São muitas e variadas as indústrias onde claramente se coloca a necessidade de otimizar a utilização de matéria-prima em processos de corte. Assim, embora com aplicações em muitos outros domínios, foi a partir das actividades industriais que inicialmente se colocou o Problema dos Cortes. De facto, frequentemente se utilizam peças (grandes), de dimensões geométricas regulares, produzidas ou então compradas, que deverão ser cortadas para obter peças (pequenas) de dimensões que satisfaçam os pedidos dos clientes ou as necessidades de outras áreas da indústria em questão. São exemplos as indústrias que trabalham com ferro, aço, papel, cartão, vidro, madeira, têxteis ou couro.

O problema, numa forma simples, consiste na selecção de combinações de cortes, a que se dá o nome de padrões de corte, de acordo com um certo conjunto de regras (as restrições do problema), que produzam as peças pedidas minimizando os desperdícios, aproveitando ao máximo as peças de matéria-prima, ou de acordo com algum outro objectivo conveniente. Ele surge naturalmente da competição internacional, e nomeadamente dos preços crescentes dos materiais usados, assim como da necessidade de desenvolver *software* sofisticado que permita a utilização de computadores na resolução deste problema, substituindo assim o planeamento manual que, baseado apenas na grande experiência dos operários que lidam com o assunto, leva muitas horas, por vezes vários dias. A estas razões acresce o facto de nalgumas indústrias, dadas as matérias-primas utilizadas, o bom aproveitamento dos materiais ser uma questão crucial.

Quando se cortam barras, tubos ou rolos, trata-se em geral de problemas a uma dimensão, pois normalmente só a dimensão comprimento é significativa. Cortando materiais planos, como chapas, placas de madeira ou têxteis, teremos cortes a duas dimensões, regulares ou mais ou menos irregulares, conforme as peças pequenas sejam rectângulos, outros polígonos ou figuras

mais ou menos complicadas. Problemas de dimensão três surgem menos frequentemente porque em muitas situações, uma ou duas das dimensões do material original podem ser estandardizadas. Ocorrem contudo sob a forma de problemas de aproveitamento de espaço no carregamento/empacotamento de peças em caixas ou contentores de dimensões fixas.

Para além do corte propriamente dito de materiais e do empacotamento de objectos, muitos outros problemas podem ser formulados como um Problema de Cortes. Do sequenciamento de memória em computadores à orçamentação de capitais, passando pela minimização da área utilizada por um conjunto de componentes num circuito VLSI e pelo planeamento dos anúncios comerciais, de modo a obter uma boa utilização do tempo disponível, são muitas e variadas as áreas onde os Problemas de Cortes surgem naturalmente. De uma forma geral, pode-se dizer que sempre que se tem um recurso valioso que se pretende dividir ou repartir em porções com tamanhos previamente determinados, tem-se um problema com a estrutura do Problema dos Cortes.

1.2 - Âmbito da dissertação.

Embora o Problema dos Cortes tenha sido apresentado pela primeira vez em 1955 por Paull e Walter, e no início dos anos 60 tenha recebido um impulso considerável a partir dos trabalhos de Gilmore e Gomory, nestes últimos 15 anos houve uma verdadeira explosão no interesse dedicado pela comunidade científica a este tipo de problemas. Para além dos diferentes cambiantes de que o problema se pode revestir, o que faz com que cada empresa seja um problema diferente, ao mesmo tempo que dificulta a criação de *software standard*, este interesse justifica-se também pela sua dificuldade intrínseca. De facto, à complexidade natural de um problema tipicamente combinatório, acrescem restrições e objectivos variados, muito dependentes das aplicações concretas e nem sempre totalmente conciliáveis. Daí que, tal como Hinxman afirmou em 1980:

"Quem se vir confrontado com um Problema de Cortes terá muito provavelmente que desenvolver o seu próprio método, a menos que a abordagem de Gilmore e Gomory seja viável."

Tais métodos terão que ser quase de certeza heurísticos. O estudo de artigos, se algum existir, sobre problemas similares pode ajudar. Contudo, os métodos heurísticos tendem a ser fortemente baseados nos problemas particulares a que são aplicados, e problemas aparentemente semelhantes podem exigir métodos heurísticos radicalmente diferentes".

Assim, várias foram as heurísticas criadas para a resolução de diferentes concretizações do Problema dos Cortes. Contudo, pouco trabalho tem sido realizado no desenvolvimento de algoritmos que forneçam soluções óptimas. Isto apesar dos modelos habitualmente usados para o Problema dos Cortes, já relaxarem algumas das restrições mais violentas, como por exemplo a integralidade das variáveis.

Nesta dissertação abordamos, numa perspectiva de optimização, um dos casos particulares do Problema dos Cortes: o problema a duas dimensões, com peças rectangulares e cortes ortogonais e em guilhotina, isto é, paralelos a um dos lados dos rectângulos e sempre de um lado a outro. O objectivo considerado será o da minimização dos desperdícios.

1.3 - Estrutura da dissertação.

Na primeira parte da dissertação, abordamos o Problema dos Cortes de uma forma genérica. Assim, no capítulo 2 é apresentada a sua estrutura geral, relacionando o problema clássico de cortar materiais com outros problemas afins. No capítulo 3 é feita uma proposta de classificação das várias formas do Problema dos Cortes, que reúne e sistematiza os trabalhos de Hinxman [24], Golden [20] e mais recentemente Dyckhoff [11], [12]. Finalmente, no capítulo 4 é apresentada uma classificação para as técnicas de resolução deste problema, com particular ênfase no caso bidimensional. Ainda neste capítulo é feito um levantamento bibliográfico, que procuramos que fosse exaustivo, de artigos que apresentam técnicas de resolução para problemas a duas dimensões. Para cada artigo é apresentado um breve resumo, tendo havido a preocupação de enquadrar a técnica apresentada dentro da classificação proposta no início do capítulo.

As abordagens ao Problema dos Cortes podem ser divididas em duas grandes classes: as técnicas uni-padrão e as técnicas multi-padrão. Sobre elas incidirão as partes II e III da dissertação, respectivamente. Esta divisão reflecte as duas formas segundo as quais o Problema dos Cortes pode surgir:

- dada uma peça grande e um conjunto de peças pequenas, pretende-se encontrar o melhor padrão de corte, isto é, a melhor maneira de dividir a peça grande em peças pequenas pertencentes ao conjunto dado. Neste caso, não se espera obter todas as peças pequenas, mas apenas aquele subconjunto que, de acordo com algum objectivo, melhor aproveita a peça grande - problemas UNI-PADRÃO;

- dado um lote de peças grandes e um conjunto de peças pequenas, pretende-se obter a forma de cortar todas as peças pequenas, minimizando o número de peças grandes gastas na satisfação de todos os pedidos de peças pequenas, e eventualmente considerando outros objectivos - problemas MULTI-PADRÃO.

É de salientar que a designação uni-padrão não significa que apenas seja considerado um padrão de corte, mas sim que a solução do problema é um padrão de corte, normalmente resultado de uma escolha entre muitos outros possíveis.

Na parte II desta dissertação são então abordadas as técnicas uni-padrão que consideram o objectivo da minimização dos desperdícios, não sendo por isso consideradas heurísticas que, por definição, não encontram ou pelo menos não garantem a solução óptima para o problema. Assim, no capítulo 5 são apresentados algoritmos que seguem uma abordagem optimizante para o Problema dos Cortes a duas dimensões com os objectivos e restrições já mencionados: o algoritmo de Christofides e Whitlock, o algoritmo de Wang e o algoritmo de Wang modificado. Este último foi por nós desenvolvido e é o resultado de algumas alterações ao algoritmo de Wang. No capítulo 6 são apresentados os resultados computacionais da comparação feita entre estes algoritmos, donde sobressai a superioridade de algoritmo de Wang modificado.

Na terceira parte desta dissertação são abordados os problemas multi-padrão e técnicas para a sua resolução. Assim, no capítulo 7 é apresentada a técnica clássica de Gilmore e Gomory, a programação linear com geração atrasada de colunas, sendo, no capítulo 8, proposta uma nova abordagem, por nós desenvolvida, a programação linear com geração rápida de colunas. Esta nova técnica, sem deixar de obter a solução ótima para o problema, ao fazer recurso a uma forma expedita de geração de colunas, consegue uma eficiência cerca de duas vezes maior que a técnica clássica, conforme se pode constatar através dos resultados computacionais da comparação das duas técnicas apresentados no capítulo 9.

Finalmente, no capítulo 10 é apresentado um breve resumo dos principais resultados obtidos, sendo também apontadas algumas ideias que poderão ser objecto de um desenvolvimento futuro. Nos apêndices A, B e C, encontram-se descrições de alguns algoritmos e heurísticas clássicos, que são usados por algumas das técnicas implementadas. Do apêndice D constam tabelas com os resultados exaustivos dos testes computacionais efectuados no âmbito do capítulo 9, uma vez que neste capítulo, por uma questão de simplicidade e clareza, apenas são apresentados os valores médios desses resultados.

O PROBLEMA DOS CORTES RECTANGULARES

Capítulo 2

O Problema dos Cortes: estrutura e relação com problemas afins.

Os problemas de cortes surgem de contextos tão variados e aparecem formulados de modos tão diversos na literatura, que se torna difícil apresentar uma definição destes problemas que, sendo correcta, seja simultaneamente simples e geral.

São vários os nomes sob os quais os problemas surgem, derivando directamente do contexto industrial ou económico de onde as aplicações concretas surgem. Assim, são usuais designações como:

- problema dos cortes, dos desperdícios ou dos aproveitamentos, nas indústrias do aço, do papel, do vidro, têxteis e outras similares;
- *bin packing*, empacotamento por riscas, empacotamento vectorial ou problema mochila, também em aplicações ligadas às indústrias já mencionadas e a problemas específicos da indústria electrónica (por exemplo, a gestão de memória por blocos e o projecto de circuitos VLSI);
- carregamento de veículos, de paletas ou contentores;
- problema de *layout*, de encaixe ou de partição, em aplicações mais predominantemente geométricas;
- aplicações de capitais, balanceamento de linhas de produção, alocação de memória e escalonamento de sistemas multiprocessador.

A literatura reflecte a dispersão destas aplicações. Podem ser encontrados trabalhos sobre o assunto em áreas diversas como as Ciências da Computação, Logística, Engenharia Industrial, Investigação Operacional, Optimização Combinatória, Produção, Matemática, Gestão, etc.

Todos estes problemas têm contudo uma estrutura com uma lógica comum:

- Existem basicamente dois grupos de dados cujos elementos definem conjuntos de figuras fixas num espaço n dimensional de números reais:

-
- por um lado um lote de objectos disponíveis que serão designados por grandes;
 - por outro lado uma lista de objectos pequenos que se pretende obter ou empacotar a partir dos objectos grandes, ou nos objectos grandes.

Os objectos grandes poderão ser chapas de aço, rolos de papel ou placas de vidro, em problemas mais especificamente de cortes, ou o espaço disponível num contentor ou numa paleta, se forem problemas ditos de empacotamento. A figura 2.1 representa ambos os grupos no caso particular do corte de chapas rectangulares, portanto num espaço a duas dimensões.

- O processo de corte (empacotamento) pressupõe padrões de corte (empacotamento), isto é, combinações geométricas de objectos pequenos afectadas a objectos grandes. Peças residuais, ou seja, figuras que surjam nos padrões (combinações de objectos pequenos associadas a um objecto grande) e que não pertençam ao conjunto dos objectos pequenos, são encaradas como desperdício.

Os problemas de empacotamento e carregamento podem ser reduzidos a problemas de cortes se se encarar o espaço disponível útil dos veículos, paletas ou contentores, como o objecto grande que se pretende cortar em bocados de espaço vazio mais pequenos, onde se encaixarão os objectos a carregar ou transportar. De facto, a forte relação entre os problemas de cortes e de empacotamento resulta da dualidade matéria-espaço, ou mais correctamente, da dualidade entre um corpo de matéria sólida e o espaço ocupado por ele.

Durante este trabalho, os problemas serão sempre designados por problemas de cortes, contudo, os conceitos e métodos de resolução apresentados são directamente aplicáveis a problemas de empacotamento similares, isto é, com as mesmas características.

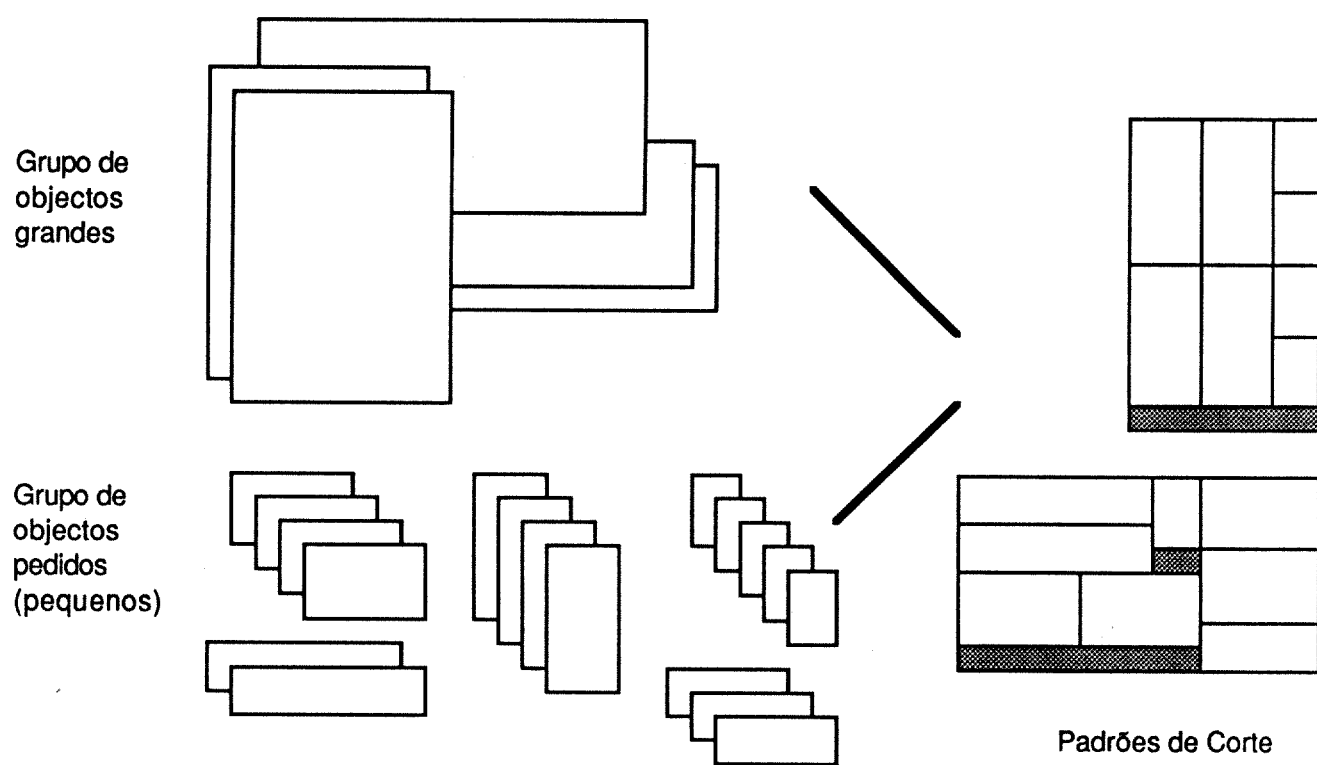


Figura 2.1 - Estrutura básica de um problema de cortes.

Capítulo 3

Classificação dos Problema de Cortes.

O objectivo de comparar diferentes formulações do problema dos cortes e suas resoluções, só pode ser alcançado se se estabelecer um conjunto de critérios adequado. Para tal é suficiente tomar os modelos matemáticos que são normalmente apresentados, embora nem sempre utilizados de facto na resolução dos problemas concretos, como base para uma sistematização, nomeadamente analisando e quantificando as permissas, dimensão, funções objectivo e restrições do modelo. Far-se-á ainda recurso a algumas questões que precedem o próprio modelo matemático, tais como as dimensões relevantes para o problema e a conjuntura de planeamento onde o problema dos cortes se insere.

Os critérios adoptados para a sistematização, por serem os que mais diferenciam os vários problemas, foram os seguintes:

- dimensão;
- conjuntura de planeamento;
- objectivos;
- restrições.

Dimensão

Dado que os problemas de cortes estão quase sempre ligados à divisão de matéria, ou do espaço ocupado por ela, os objectos ligados ao processo de corte são tri-dimensionais. Contudo, por vezes apenas uma ou duas dimensões são relevantes para o problema. Pode-se então diferenciar os problemas com sendo:

- a uma dimensão;
 - a duas dimensões;
 - tri-dimensionais.
-

Exemplos típicos de problemas a uma dimensão, encontram-se em linhas de produção onde barras ou rolos de matéria-prima têm que ser cortados em peças mais pequenas com a mesma secção transversal. Neste caso, apenas uma dimensão do material (o comprimento) é relevante para a resolução do problema dos cortes, uma vez que a secção transversal é fixa e não afecta a solução do problema. Existem ainda problemas que são intrinsecamente uni-dimensionais, ou seja, em que os objectos têm apenas uma dimensão: é o caso da divisão de períodos de tempo em intervalos mais pequenos.

Problemas a duas dimensões surgem quando material plano (por exemplo, chapas de metal, folhas de vidro, têxteis ou placas de circuitos electrónicos) tem que ser dividido em peças com áreas menores, mas com a mesma espessura (dimensão não relevante para o problema).

Os problemas tri-dimensionais são mais raros, pois em muitas situações reais uma ou duas dimensões da matéria-prima podem ser estandardizadas, reduzindo assim a dimensionalidade do problema. Isso não acontece em problemas de carregamento, caso particular do problema de empacotamento, em que se pretende alocar um certo conjunto de objectos, de tamanho conhecido, a caixas de capacidade limitada, de forma a minimizar o número de caixas utilizadas. Nestes casos o problema dos cortes tem que ser encarado como um problema tri-dimensional.

Pode-se considerar uma subdivisão destas classes de problemas, quando adicionalmente aos graus de liberdade existentes na geração de padrões, uma das dimensões relevantes do problema não está unicamente determinada, ou seja, é variável. Um exemplo típico desta situação é o corte de figuras bi-dimensionais de um rolo de tecido, onde a largura é fixa mas o comprimento disponível variável (basta desenrolar mais o rolo para obter mais comprimento). O objectivo do problema seria minimizar o comprimento total de tecido necessário para satisfazer os pedidos. Problemas com duas dimensões relevantes, em que uma é fixa e a outra variável, são habitualmente designados por problemas de dimensão 1.5. De modo semelhante se poderão definir problemas de dimensão 2.5 como problemas em que duas das dimensões são fixas e uma terceira dimensão é variável. É de realçar que estes problemas são intrinsecamente problemas de

dimensão 1 e 2, respectivamente, pois a função objectivo depende unicamente de uma ou duas das dimensões do problema.

A tabela 3.1 sintetiza as possíveis classificações, quanto à dimensão, dos problemas de cortes.

Tabela 3.1 - Dimensionalidade do problema dos cortes

Dimensionalidade do Problema	Problema	Material original e o resultado dos cortes coincidem na:
1 dimensão	Divisão de um comprimento dado em tamanhos pedidos	Secção transversal e qualidade
1 dimensão e meia	Divisão de uma folha de comprimento variável em tamanhos pedidos	Espessura e qualidade
2 dimensões	Divisão de uma dada folha em tamanhos pedidos	Espessura e qualidade
2 dimensões e meia	Divisão de um volume de altura variável em tamanhos pedidos	Qualidade
3 dimensões	Divisão de um dado volume em tamanhos pedidos	Qualidade

Conjuntura de planeamento

Numa situação real de produção, as decisões afectam um todo que exige uma optimização global e completa, onde a optimização do processo de corte se integra. Uma vez que existem muitas interdependências entre as várias áreas de produção, se se pretende um óptimo global para o processo de produção total, é necessário definir uma conjuntura de planeamento para o problema dos cortes, onde se perca o menor número de interdependências possível. A optimização do

processo de corte pode sobretudo interferir com questões de sequenciamento de máquinas, planeamento de inventários e armazenagens, e tempos de produção. Pode ainda, eventualmente, ter influência na selecção de processos de fabrico alternativos (por exemplo, na selecção de máquinas diferentes). Na figura 3.1 estas relações são representadas.

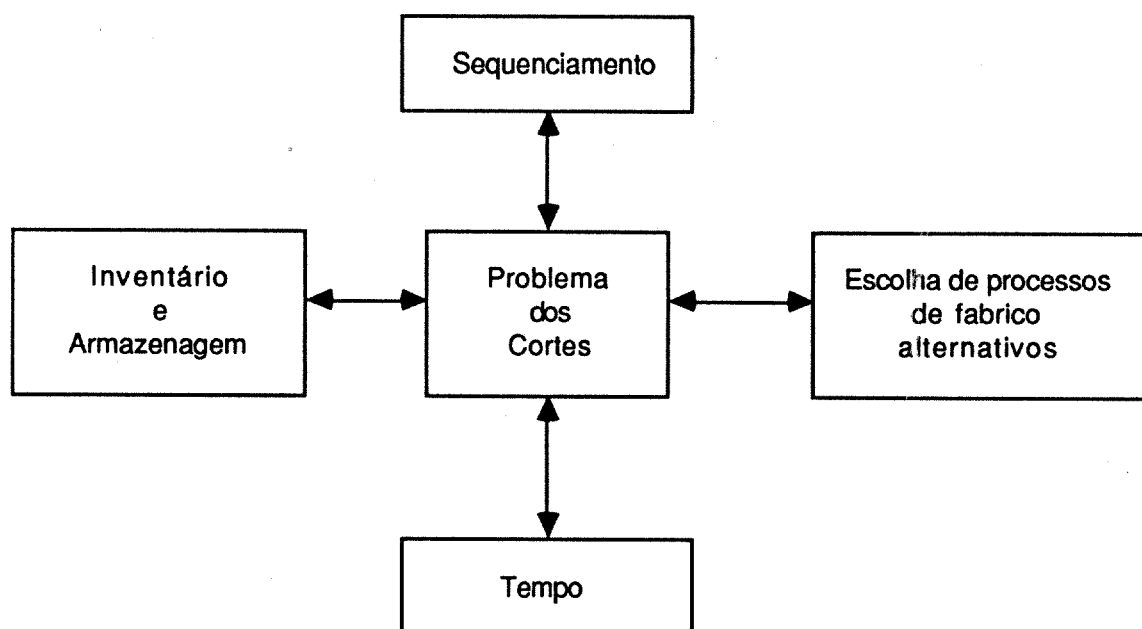


Figura 3.1 - Interdependência entre o problema dos cortes e outros problemas de produção

Assim, o planeamento simultâneo é fundamental para que se obtenha um óptimo global teórico. Contudo, na maioria dos casos isto levaria a conceitos que não podem ser implementados com as soluções técnicas disponíveis, ou pelo menos dariam origem a um esforço computacional proibitivo. Existem então, basicamente, duas possibilidades para um procedimento prático:

- um tratamento sequencial de todas as áreas de produção, em que a solução de um problema determina as restrições para o problema subsequente. Neste caso o problema dos cortes é tratado isoladamente;
- integrar alguns problemas num planeamento simultâneo com o problema dos cortes, desde que o esforço dispendido na obtenção da solução se mantenha dentro de limites aceitáveis.

Objectivos

A transformação de um problema de cortes num modelo matemático de decisão, requer sempre a construção de uma função objectivo. Mesmo aquelas técnicas de resolução que não utilizam nenhum modelo matemático formal, têm que seleccionar um objectivo para o problema, pelo que a expressão função objectivo deverá ser entendida num sentido bastante lato, de forma a abranger estas situações.

A bem da qualidade do modelo, é fundamental que a função objectivo reflecta o melhor possível, os motivos e necessidades concretas que levaram a equacionar e tentar resolver o problema dos cortes. Assim, a função objectivo a usar no modelo do problema dos cortes, depende em primeiro lugar do principal objectivo da estrutura organizacional onde se insere o processo de corte. Em segundo lugar, a formulação dependerá da porção de realidade considerada, podendo aqui haver uma separação entre os casos em que o problema é considerado isoladamente e os casos em que se tenta o planeamento e optimização de vários aspectos em simultâneo.

No caso da optimização isolada do processo de corte, pode-se habitualmente ignorar a aplicação directa de considerações do tipo monetário, tais como custos, lucros ou valores acrescentados. Normalmente os objectivos são traduzidos em termos de minimização da quantidade de sucata (minimização de desperdícios) ou da quantidade de material usado (minimização das entradas). Quando os pedidos são exactamente satisfeitos, então os dois objectivos são equivalentes, no sentido de que a solução que origina desperdício mínimo também origina um gasto mínimo de peças de matéria-prima (entrada do processo de corte), e vice-versa. Ambos os objectivos são casos particulares do objectivo "maximização da productividade", que pode ser expresso na variável "desperdício relativo" como sendo a proporção sucata/material total usado. Estes objectivos, se bem que simples e usados na esmagadora maioria das aplicações reais, são de certa forma limitativos, uma vez que atribuem o mesmo valor à unidade de tamanho do material original usado e ao que é considerado desperdício, não considerando uma possível

reutilização deste material, ou se o considera, não diferencia diferentes estruturas ou formas, ignorando portanto que peças com um mesmo peso mas diferentes dimensões não têm necessariamente as mesmas hipóteses de reutilização. Nestes casos seria aconselhável atribuir preços por unidade de tamanho diferentes, conforme o tamanho e forma do material original, considerando ainda o caso em que o material de entrada já é a sobra de um processo de corte anterior.

O planeamento simultâneo de várias áreas do processo produtivo, implica a combinação de objectivos individuais, ligados a cada uma das áreas de planeamento, numa só função objectivo, a menos que seja usada qualquer uma das técnicas da optimização multi-critério. Na prática estas técnicas não têm sido aplicadas, optando-se usualmente por heurísticas que tentam pesar de algum modo cada um dos diferentes objectivos envolvidos.

Restrições

A forma mais apropriada para a classificação das restrições que afectam o problema dos cortes, surge quando o processo de corte é modelizado como processo de entrada/saída (ver figura 3.2). Este modelo descreve a transição dos materiais de entrada (factores de produção) para os materiais de saída (produtos), assim como o processo tecnológico que determina a transição. Estes três elementos do modelo podem ser descritos em termos de quantidade e valor, originando assim três tipos de restrições:

- (i) - numa situação de planeamento real, a matéria-prima (materiais de entrada) só estão disponíveis em quantidades limitadas e em determinados tamanhos - restrições de entrada.
 - (ii) - a tecnologia de corte não pode ser mudada sempre que e quando se queira. Por exemplo, a parte da tecnologia que é definida pelos parâmetros das máquinas de corte, é limitada aos valores admissíveis para esses parâmetros - restrições às operações de corte.
 - (iii) - também a quantidade de material de saída a produzir, definida em termos das várias qualidades e tamanhos, está sujeita a diversas restrições, que são devidas aos pedidos existentes no momento, quer internos quer externos - restrições de saída.
-

Para além destas restrições quantitativas, poderiam ainda ser consideradas restrições qualitativas ou de custos. De facto é possível que orçamentos pré-definidos ou valores acrescentados mínimos a obter, limitem a liberdade de planeamento e de acção. Contudo, na prática as contribuições económicas são consideradas apenas nas funções objectivo dos modelos, razão pela qual estas restrições não serão subsequentemente aprofundadas.

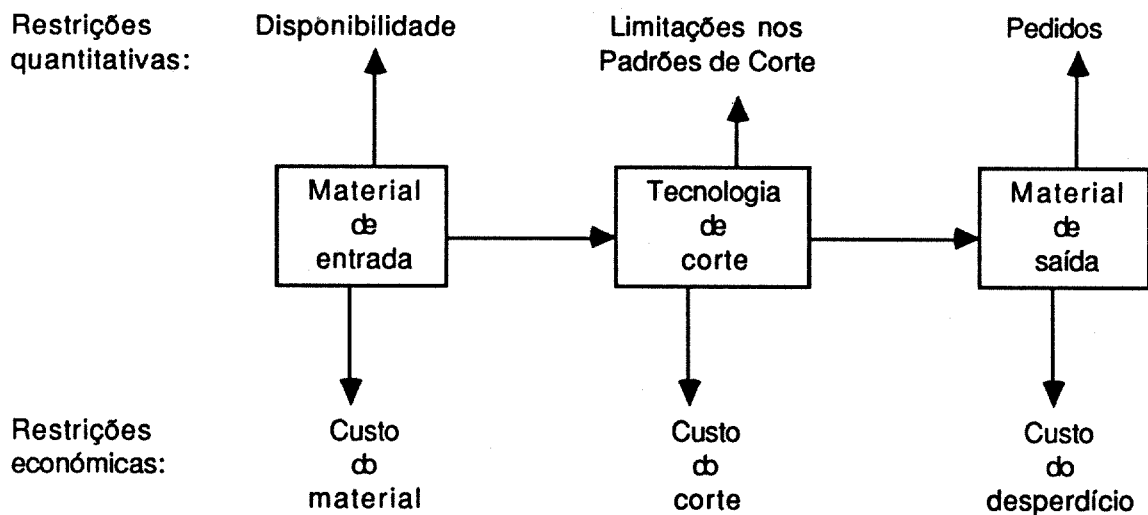


Figura 3.2 - Estrutura do problema dos cortes num modelo entrada/saída

Restrições de entrada

Geralmente todo o tipo de material de entrada, com um dado tamanho e uma dada qualidade, só está disponível em quantidades limitadas, razão pela qual esta restrição deveria ser sempre considerada. Contudo, por vezes o material de entrada pode ser obtido num espaço de tempo muito curto, ou existe em quantidades suficientemente grandes face às quantidades necessárias para um dado horizonte de planeamento, de forma que esta restrição pode ser abandonada.

As restrições de entrada que resultam das propriedades geométricas dos objectos a cortar, são de uma grande importância para a construção do modelo matemático, uma vez que estas restrições influenciam a geração dos padrões de corte. Neste aspecto em particular, pode-se

fazer a seguinte distinção: são todos os objectos que constituem o material de entrada do mesmo tamanho ou deve o modelo considerar diferentes tamanhos para a entrada? Se só existem objectos do mesmo tamanho então só é necessário gerar padrões de corte para este tamanho, senão devem ser gerados padrões para cada um dos tamanhos que possam surgir na entrada do processo de corte. Dependendo ainda se estes tamanhos são ou não conhecidos no início do período de planeamento, poderá ainda o modelo ter que lidar com tamanhos determinísticos ou aleatórios. O caso de tamanhos aleatórios pode surgir quer de materiais retirados de armazenamento, quer de materiais gerados numa linha de produção contínua, onde o processo de corte se insere. Representativos destas duas hipóteses são os casos de rolos de filme retirados de armazém, que por vezes têm os bordos deteriorados, originando uma largura aleatória para o rolo, e o de um processo de fabrico contínuo de rolos de chapa, cuja largura não pode ser produzida com exactidão.

Há ainda a considerar a distinção entre o caso em que o material de entrada surge sempre em tamanhos standardizados, e o caso em que juntamente com esses tamanhos surgem outros, que poderão ser por exemplo desperdícios de processos de corte anteriores.

Na prática, os diversos tipos de restrições de entrada apresentados, não conduzem a diferentes modelos, pois é habitual simplificar a formulação das restrições de modo ao problema se tornar viável, isto é, de modo a ser possível obter uma solução em tempo aceitável. Por exemplo, o caso de entradas aleatórias não é tratado na literatura, sendo sempre reduzido ao caso determinístico.

Restrições às operações de corte

Os condicionalismos que afectam as operações de corte, têm a ver exclusivamente com a admissibilidade e adequação dos padrões de corte. Sendo assim, podemos designar as restrições às operações de corte como restrições aos padrões de corte. Estas restrições resultam

fundamentalmente de exigências quanto a aspectos organizacionais, do material e de condicionalismos da tecnologia de corte em si mesma.

As restrições devidas a aspectos organizacionais são impostas, não pelo processo de corte, mas por outras áreas de produção ou planeamento. Um exemplo típico é o da obrigatoriedade de um conjunto de pedidos de um dado cliente ser satisfeito o mais rapidamente possível, implicando que se corte em primeiro lugar os seus pedidos. Este tipo de restrições não surge nas variáveis do problema mas limita de facto a geração dos padrões de corte.

As restrições atribuíveis ao tipo de material que está a ser cortado, resultam basicamente da sua estrutura e propriedades físicas. Reflectem-se por exemplo na possibilidade ou não de alterar a orientação dos objectos pedidos, dependendo da isotropia do material, ou na deformação introduzida nos objectos cortados, que terá que ser tida em consideração mantendo uma distância mínima entre as formas geométricas no padrão de corte.

As condições tecnológicas também impõem restrições devidas às propriedades especiais das diferentes ferramentas de corte. É necessário contudo, distinguir entre um processo de corte que origina só figuras rectangulares (a duas ou mais dimensões) e um processo de corte que tenha que lidar com figuras irregulares. A indústria têxtil é um exemplo típico deste último caso. A existência de figuras irregulares implica uma tecnologia de corte por *laser*, oxi-acetileno, ou outra que permita uma deslocação com tantos graus de liberdade quantas as dimensões relevantes do problema. Esta tecnologia de corte não introduz restrições adicionais aos padrões de corte.

No corte de figuras rectangulares, é necessário fazer mais algumas distinções. A primeira delas é referente à orientação dos cortes: se estes são feitos sempre paralelamente a um dos lados do objecto a cortar, diz-se que o padrão de corte é ortogonal; se os cortes podem ter uma orientação qualquer, diz-se que o padrão é não-ortogonal. A figura 3.3 apresenta um exemplo de cada um destes tipos de padrões.

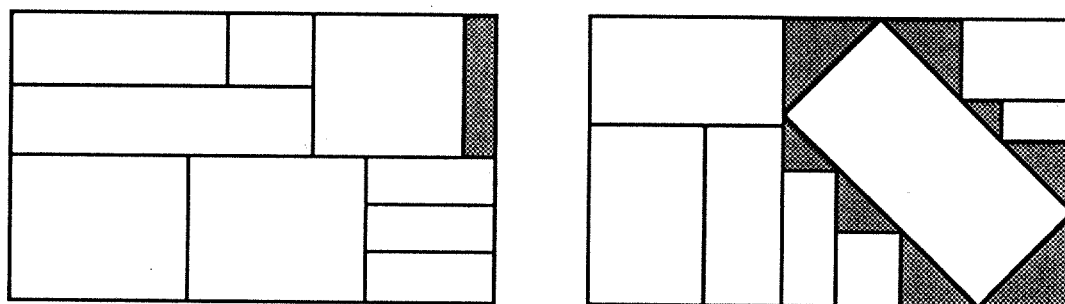


Figura 3.3 - Padrões de corte ortogonal e não ortogonal

Os padrões ortogonais podem ainda ser divididos em duas classes. Se a tecnologia de corte é tal que os cortes são sempre executados de um lado a outro do material, como seria o caso do corte de folhas de papel com uma guilhotina, então diz-se que são padrões de corte do tipo guilhotina. Em oposição, se se permitirem descontinuidades no corte em qualquer posição, poderemos ter outros tipos de padrões que não em guilhotina. A figura 3.4 ilustra estes conceitos.

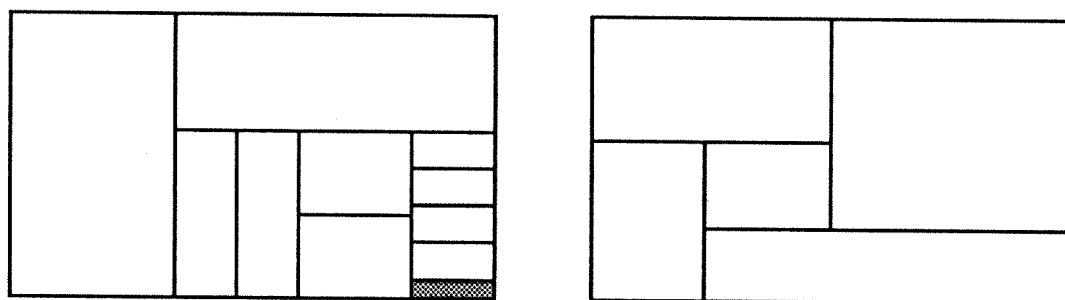


Figura 3.4 - Padrão tipo guilhotina e padrão não "guilhotinável"

Na literatura surgem ainda diferenciações dentro da classe dos padrões tipo guilhotina. Normalmente os padrões podem ter um número arbitrário de cortes, lado a lado, paralelos. Há contudo casos em que existem limites, devido a restrições no número de lâminas de corte disponíveis.

Finalmente, os problemas são por vezes classificados como em n fases, significando que para que um padrão possa ser executado, são necessárias $n-1$ mudanças de orientação das lâminas de corte. A figura 3.5 representa um padrão de corte tipo guilhotina em 3 fases.

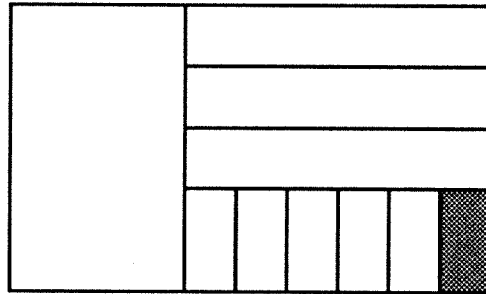


Figura 3.5 - Padrão de corte tipo guilhotina em 3 fases

Restrições de saída

O resultado do processo de corte, os objectos pequenos, tem que ser compatível, no tipo de objectos e na sua quantidade, com os pedidos existentes, que normalmente correspondem a encomendas de clientes. A primeira consideração a fazer tem a ver com os tamanhos dos objectos cortados. É razoável admitir que se o cliente pede um conjunto de objectos com determinadas dimensões, estas têm que ser respeitadas. Contudo, nalgumas indústrias são admissíveis pequenos desvios, como por exemplo na indústria do papel, onde é normal aceitarem-se variações de $\pm 5\%$.

Outro tipo de restrição diz respeito às quantidades produzidas de cada tipo de objecto. Aqui não se pode afirmar que exista uma norma, pois as situações variam extremamente:

- há casos em que as encomendas têm que ser exactamente satisfeitas - é o que vulgarmente se designa por problema dos cortes restrito;
- noutras situações são admissíveis sobreproduções, ou porque o cliente as aceita, ou porque as quantidades em excesso podem ser armazenadas;

- por vezes admitem-se também subproduções, ou seja, pretendem-se n objectos de um dado tipo, $\pm x\%$;
- finalmente, podem não existir quaisquer restrições ao número de objectos produzidos - o problema dos cortes não restrito.

Na tabela 3.2 apresenta-se uma sistematização dos critérios de classificação que na prática são considerados nos modelos do problema dos cortes, e que servirão de base à definição dos tipos de problemas que cada uma das técnicas, que serão apresentadas ao longo desta dissertação, é capaz de resolver. Não será considerado o caso em que o problema dos cortes é tratado em conjunto com outros problemas de produção, uma vez que em implementações reais tal não é habitualmente feito.

Tabela 3.2 - Classificação do problema dos cortes

Problema de cortes ...

... de dimensão: 1, 1 1/2, 2, 2 1/2, 3.																							
... com objectivo de ...	... minimizar o desperdício. ... minimizar a quantidade de matéria-prima gasta.																						
... com restrições no(s)	<table> <tr> <td>... material de entrada que ...</td><td> ... existe em quantidades ilimitadas. ... existe em quantidades limitadas. </td></tr> <tr> <td></td><td> ... tem um só tamanho. ... tem vários tamanhos. </td></tr> <tr> <td></td><td> ... inclui reaproveitamentos de processos de corte anteriores. ... só inclui medidas estandardizadas. </td></tr> <tr> <td>... padrões de corte ...</td><td> ... onde são permitidas rotações das figuras. ... onde a orientação das figuras é fixa. </td></tr> <tr> <td></td><td> ... que incluem figuras irregulares. ... que só têm rectângulos ... </td></tr> <tr> <td></td><td> ... colocados não ortogonalmente. ... colocados ortogonalmente ... </td></tr> <tr> <td></td><td> ... cortados arbitrariamente. ... cortados em guilhotina... </td></tr> <tr> <td></td><td> ... com ou sem limitações no número de lâminas e fases do corte. </td></tr> <tr> <td>... material de saída que ...</td><td> ... tem que ter exactamente as dimensões pretendidas. ... pode variar as dimensões dentro de uma dada tolerância. </td></tr> <tr> <td></td><td> ... não tem restrições nas quantidades produzidas de cada tipo. ... tem que ser produzido exactamente nas quantidades pedidas. </td></tr> <tr> <td></td><td> ... pode ser produzido com excedentes. ... pode ser produzido com excedentes ou falhas, dentro de uma tolerância dada. </td></tr> </table>	... material de entrada que ...	... existe em quantidades ilimitadas. ... existe em quantidades limitadas.		... tem um só tamanho. ... tem vários tamanhos.		... inclui reaproveitamentos de processos de corte anteriores. ... só inclui medidas estandardizadas.	... padrões de corte ...	... onde são permitidas rotações das figuras. ... onde a orientação das figuras é fixa.		... que incluem figuras irregulares. ... que só têm rectângulos ...		... colocados não ortogonalmente. ... colocados ortogonalmente ...		... cortados arbitrariamente. ... cortados em guilhotina...		... com ou sem limitações no número de lâminas e fases do corte.	... material de saída que ...	... tem que ter exactamente as dimensões pretendidas. ... pode variar as dimensões dentro de uma dada tolerância.		... não tem restrições nas quantidades produzidas de cada tipo. ... tem que ser produzido exactamente nas quantidades pedidas.		... pode ser produzido com excedentes. ... pode ser produzido com excedentes ou falhas, dentro de uma tolerância dada.
... material de entrada que ...	... existe em quantidades ilimitadas. ... existe em quantidades limitadas.																						
	... tem um só tamanho. ... tem vários tamanhos.																						
	... inclui reaproveitamentos de processos de corte anteriores. ... só inclui medidas estandardizadas.																						
... padrões de corte ...	... onde são permitidas rotações das figuras. ... onde a orientação das figuras é fixa.																						
	... que incluem figuras irregulares. ... que só têm rectângulos ...																						
	... colocados não ortogonalmente. ... colocados ortogonalmente ...																						
	... cortados arbitrariamente. ... cortados em guilhotina...																						
	... com ou sem limitações no número de lâminas e fases do corte.																						
... material de saída que ...	... tem que ter exactamente as dimensões pretendidas. ... pode variar as dimensões dentro de uma dada tolerância.																						
	... não tem restrições nas quantidades produzidas de cada tipo. ... tem que ser produzido exactamente nas quantidades pedidas.																						
	... pode ser produzido com excedentes. ... pode ser produzido com excedentes ou falhas, dentro de uma tolerância dada.																						

Capítulo 4

Abordagens à resolução do Problema dos Cortes rectangulares.

4.1 - Classificação das técnicas.

As técnicas para a resolução de problemas de cortes, podem, de uma forma geral, ser divididas em dois grupos: as técnicas algorítmicas, que se caracterizam por determinar sempre a solução óptima para o modelo proposto, e as técnicas heurísticas, que determinam uma solução, que sendo suficientemente boa para a aplicação em causa, não é garantidamente a solução óptima. Se por um lado existem relativamente poucas técnicas algorítmicas para a resolução deste problema, uma vez que o seu custo computacional é elevado, por outro lado as técnicas heurísticas são em número elevado, tendo como característica comum serem suficientemente rápidas e boas (no sentido de satisfazerem de uma forma aceitável a função objectivo). De facto, como Hinxman [24] afirma:

"O ponto da situação é tal que quem se deparar com um problema de cortes terá provavelmente que desenvolver o seu próprio método, a menos que o método de Gilmore e Gomory seja adequado. Tais métodos estão praticamente limitados a serem heurísticas. O estudo de artigos, se existirem, sobre problemas semelhantes pode ser útil. Contudo, os métodos heurísticos tendem a ser fortemente baseados nos problemas concretos a que são aplicados, e problemas aparentemente semelhantes podem requerer soluções heurísticas radicalmente diferentes."

A aproximação clássica para a resolução destes problemas (apresentada por Gilmore e Gomory) caracteriza-se por um procedimento em duas fases. Numa primeira fase são gerados padrões de corte adequados, por enumeração, muitas vezes usando heurísticas. Em seguida resolve-se um problema de Programação Linear para determinar o número de vezes que cada padrão de corte é utilizado. Se na fase da geração de padrões a enumeração for total, ou seja, se se gerarem todos os padrões de corte possíveis, e se usar Programação Linear Inteira, então esta técnica será do tipo algorítmico - originará a solução óptima.

A enumeração total dos padrões de corte pode ser evitada se se usar a técnica de Programação Linear com geração atrasada de colunas [16], cuja aplicação ao problema dos cortes foi proposta também por Gilmore e Gomory. Esta técnica consiste basicamente em gerar as colunas (que como mais tarde se verá, representam padrões de corte) do problema de Programação Linear, à medida que as respectivas variáveis forem sendo seleccionadas para entrar na base do quadro Simplex. Qualquer outra aproximação ao problema, como sendo a de gerar apenas alguns bons padrões e depois aplicar PL a esse conjunto, ou relaxar a integralidade do problema, coloca esta técnica na classe das técnicas heurísticas.

As técnicas heurísticas usam um conjunto de conceitos e metodologias que na essência não variam muito e de que em seguida se apresentam os mais significativos:

- Na técnica da pesquisa no espaço dos estados, potenciais soluções parciais do problema são consideradas como nós de um grafo. Faz-se então uma pesquisa sobre esse grafo, desde o nó inicial, que corresponde ao problema completamente por resolver, até ao nó final, que corresponde ao problema completamente resolvido. Esta técnica é semelhante a um algoritmo de pesquisa em árvore, apenas diferindo deste por não considerar todos os nós da árvore de pesquisa, mas somente alguns deles, não havendo sequer a garantia de que não se esteja a excluir o nó óptimo.
 - Num método baseado na redução do problema, o problema inicial é decomposto em sub-problemas mais pequenos, que por sua vez são eventualmente decompostos outra vez. A união das soluções de todos os sub-problemas resultantes da decomposição forma a solução do problema original.
 - Um algoritmo iterativo pode ser transformado numa técnica heurística, forçando a sua paragem antes que todas as iterações tenham sido efectuadas. O critério de paragem pode ser o custo computacional das iterações efectuadas até ao momento, ou os valores das soluções produzidas por iterações consecutivas diferirem entre si menos que uma certa quantidade. Nestes casos diz-se que se está perante uma heurística do tipo cut-off.
-

- Quando se está a gerar um conjunto de soluções admissíveis para o problema, pode-se decidir aceitar aquelas que satisfaçam um certo critério. Nesta situação diz-se que se usou um nível de aspiração para determinar a aceitação ou não de uma solução.
 - Uma técnica comum a muitas heurísticas consiste em, uma vez gerado um padrão de corte, aplicá-lo o maior número de vezes possível (sem causar sobreproduções, se for o caso disso). O conjunto de pedidos é então reduzido de todos os que foram satisfeitos pela aplicação repetida desse padrão, sendo em seguida gerado um novo padrão, que por sua vez será usado tantas vezes quantas as possíveis. A este método, que pertence à classe dos métodos de redução do problema (a redução acontece na lista de pedidos), dá-se o nome de redução por repetição exaustiva.
 - A cada objecto cortado está associado um valor, que geralmente é proporcional ao comprimento, área ou volume do objecto. Por vezes um valor diferente pode ser atribuído. Este valor, determinado heurísticamente, poderá por exemplo reflectir o desejo de que o objecto seja incluído nesse padrão de corte, uma vez que por uma questão de satisfação rápida de um dado pedido, ou por questões ligadas ao sequenciamento dos padrões de corte, pode haver interesse em que um objecto cortado de um padrão seja também cortado do padrão seguinte. Esta estratégia de valorização heurística pode também ser aplicada aos objectivos do problema: quando existe mais que um objectivo, sendo alguns antagónicos, o ponto de equilíbrio da satisfação simultânea dos vários objectivos, pode ser aproximado pela atribuição de pesos a cada um dos objectivos e consequente fusão num só.
 - Outro conjunto de heurísticas funciona por amostragem. Basicamente consiste na geração de um certo número de pontos do espaço das soluções admissíveis e no cálculo do valor da função objectivo para cada um desses pontos. O ponto a que corresponder maior valor é escolhido para solução do problema.
 - Existe ainda um conjunto de técnicas heurísticas que simplesmente colocam as peças segundo um ordem pré-estabelecida e uma estratégia de colocação previamente definida. Aliás, é a
-

estratégia, ou estratégias, de colocação que diferenciam estas heurísticas. Dada a aplicação preferencial que têm tido, são habitualmente designadas por heurísticas de empacotamento.

Finalmente pode-se dividir as técnicas de resolução do problema dos cortes em duas outras grandes classes:

- os métodos que atacam o problema global, resolvendo simultaneamente todos os pedidos e gerando todos os padrões de corte necessários - métodos multi-padrão;
- os métodos que geram um padrão de corte de cada vez, procurando o padrão seguinte resolver os pedidos não satisfeitos pelo padrão anterior - métodos uni-padrão.

4.2 - Levantamento bibliográfico

Apresenta-se em seguida um resumo dos artigos publicados sobre o problema dos cortes rectangulares ortogonais a duas dimensões. Foram considerados todos os artigos que apresentassem uma técnica ou abordagem diferentes, assim como aqueles que contivessem testes ou comparações entre técnicas. O autor desta dissertação procurou ser exaustivo nesta pesquisa, considerando todos os trabalhos publicados em revistas de língua inglesa constantes dos "International Abstracts in Operations Research". Contudo, não fica excluída a hipótese de existir algum trabalho não mencionado no presente levantamento bibliográfico.

Gilmore e Gomory, 1965 [18]

"Multistage Cutting Stock Problems of Two and More Dimensions"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimização do número de objectos grandes.

Restrições: Cortes em guilhotina;

Número de rectângulos de cada tipo restrito (sem sub ou sobreproduções).

Os autores começam por fazer uma revisão do modo como abordaram o problema a uma dimensão, apresentando em seguida uma generalização para o problema a duas dimensões. Para tal apresentam a seguinte formulação PL:

$$\begin{aligned} \min \quad & \sum_i X_i \\ \text{sujeito a:} \quad & \sum_i a_{ij} X_i = b_j \\ & X_i \geq 0 \text{ e } X_i \text{ inteiro} \end{aligned}$$

em que: $[a_{ij}]$ - padrão de corte i . Cada componente traduz o número de rectângulos do tipo j que o padrão produz;

b_j - número de rectângulos do tipo j que devem ser produzidos;

X_i - número de vezes que o padrão i deve ser usado.

No modelo surge como objectivo a minimização do número de vezes que os padrões, no conjunto, são utilizados, mas este é um objectivo equivalente ao da minimização do número de chapas de *stock* (material de entrada) a gastar. Neste caso, em que não são permitidas nem sub nem sobre produções, estes objectivos ainda são equivalentes a um terceiro, o da minimização do desperdício.

Os autores admitem depois o relaxamento do problema, isto é, consideram apenas $X_i \geq 0$. Afirmam que a formulação como PL Inteira é impraticável e que o arredondamento das soluções origina habitualmente boas soluções. Contudo, mesmo com esta simplificação, o modelo continua impraticável, uma vez que o número de padrões possíveis de construir, mesmo para problemas pequenos, é demasiado elevado. Os autores ultrapassam esta questão através do método da geração atrasada de colunas. Usando esta técnica, o problema PL arranca apenas com algumas colunas, gerando outras à medida que tal se torna necessário. Esta necessidade é quantificada através da resolução de um problema auxiliar em cada iteração do método Simplex: um problema tipo mochila.

A técnica apresentada neste artigo é habitualmente considerada e designada como a técnica clássica para a resolução do problema dos cortes. Apesar das limitações inerentes ao modelo usado, esta técnica enquadra-se na classe das técnicas algorítmicas, uma vez que determina a solução óptima para o modelo que se propõe.

Hahn, 1968 [21]

"On the Optimal Cutting of Defective Sheets"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Minimização do desperdício.

Restrições: Folhas de *stock* com defeitos;

Cortes em guilhotina;

Cortes em 3 fases.

A técnica apresentada neste artigo gera o padrão de corte óptimo, para um dado conjunto de pedidos e considerando um folha de *stock* com áreas defeituosas, minimizando o desperdício. As áreas defeituosas são consideradas com a forma de rectângulos.

O algoritmo começa, numa primeira fase, por cortar paralelamente ao eixo dos xx , obtendo secções. Em seguida as secções são cortadas, paralelamente ao eixo dos yy , em riscas e finalmente as riscas nos rectângulos pedidos. É usada Programação Dinâmica para colocar os rectângulos, sendo-lhes atribuído diferentes valores, conforme a sua utilidade (portanto não proporcionais às respectivas áreas). Basicamente o problema a duas dimensões é decomposto em 3 problemas a uma dimensão, correspondendo um a cada fase de corte.

Se o problema a resolver não for intrinsecamente um problema em três fases, então esta técnica deve ser encarada como uma heurística de redução do problema - ver classificação proposta em 4.1.

Herz, 1972 [23]

"Recursive Computational Procedure for Two-Dimensional Stock Cutting"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Minimização do desperdício.

Restrições: Cortes em guilhotina;
Número de rectângulos não restrito.

O algoritmo apresentado baseia-se num procedimento recursivo que vai sucessivamente cortando os rectângulos provenientes de cortes anteriores. Quando não é possível fazer mais cortes o algoritmo retrocede, desfazendo os últimos cortes, e tenta novas hipóteses. Fundamentalmente este é um algoritmo de pesquisa em árvore.

Para diminuir o número dos cortes que são testados, é feita uma redução do espaço das coordenadas onde é admissível fazer um corte, a partir de considerações de acessibilidade dessas coordenadas por combinações lineares dos rectângulos a cortar e de considerações de simetria. A eficiência computacional é ainda aumentada através da memorização das partes da árvore de pesquisa já exploradas, para que, se o mesmo rectângulo voltar a surgir, os cálculos não sejam repetidos.

O autor compara ainda a eficiência deste algoritmo com o de Gilmore e Gomory [19] que resolve o mesmo problema, concluindo ser o algoritmo recursivo 20% mais rápido que o algoritmo iterativo de Gilmore.

Dyson e Gregory, 1974 [13]

"The Cutting Stock Problem in the Flat Glass Industry"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimização do desperdício;
Sequenciamento adequado dos padrões de corte.

Restrições: Cortes em guilhotina;
Número de rectângulos restrito.

Neste artigo é apresentada uma aplicação real do problema dos cortes a um processo de produção da indústria do vidro. Face ao problema clássico, esta aplicação introduz a complicação adicional do sequenciamento dos padrões de forma a minimizar-se o número de descontinuidades na satisfação das encomendas.

Os autores começaram por aplicar o método de Gilmore e Gomory, sequenciando depois os padrões obtidos de forma a minimizar o número de descontinuidades. Contudo, esta aplicação mostrou-se inviável, dado o tempo necessário à geração do conjunto de padrões com desperdício mínimo, assim como as restrições ao número de pedidos e padrões que os programas impunham.

Como consequência, desenvolveram uma heurística que produz padrões sequencialmente, segundo a ordem pela qual deverão ser executados. A heurística consiste fundamentalmente em:

- gerar o melhor padrão de corte usando a técnica da função mochila de Gilmore e Gomory;
- repetir esse padrão até que a quantidade pedida de um dos rectângulos, que figure no padrão, seja atingida;
- repetir o processo para os pedidos ainda não satisfeitos.

Para diminuir ao número de descontinuidades e aumentar os níveis de aproveitamento da matéria-prima, os autores introduziram os seguintes melhoramentos na heurística descrita:

- aumentar o valor dos rectângulos que se preveja virem a originar um grande desperdício se forem cortados ao fim;
-

- aumentar o valor dos rectângulos que estejam presentes no padrão anterior e cujos pedidos não tenham sido completamente satisfeitos.

A aplicação desta heurística, se bem que ultrapassando os problemas postos pelo algoritmo de Gilmore e Gomory, não conseguiu resultados significativamente melhores que os obtidos por um trabalhador experimentado, através de colocação manual.

Esta heurística utiliza simultaneamente uma estratégia de redução por repetição exaustiva, ao repetir um padrão enquanto tal for possível, e uma estratégia de valorização heurística dos rectângulos a cortar (ver 4.1).

Adamowicz e Albano, 1976 [1]

"A Solution of the Rectangular Cutting-Stock Problem"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Minimização do desperdício;

Restrições: Número de rectângulos restrito.

Os autores apresentam uma heurística para problemas de grande dimensão (elevado número de rectângulos a cortar), baseada na actuação de operadores experimentados na geração manual de padrões de corte.

A heurística começa por escolher a folha, ou sub-folha (parte da folha), que se deve tentar preencher. Em seguida, gera riscas (grupos de rectângulos com uma dimensão em comum) e selecciona um grupo de riscas para preencher a folha correntemente sob consideração. Se a solução obtida não for satisfatória, é determinada uma divisão diferente da folha e o processo relançado. Se for satisfatória, o padrão é guardado, os desperdícios determinados e uma nova folha escolhida. O procedimento termina quando não houver mais rectângulos para colocar.

A eficiência da heurística depende sobretudo da forma segundo a qual são escolhidas as riscas que vão preencher uma dada folha ou sub-folha. Para tal, os autores usam um algoritmo de Programação Dinâmica a uma dimensão.

No final, as zonas de desperdício são ainda consideradas como hipotéticas zonas de colocação.

Fundamentalmente, esta heurística, ao decompor um problema a duas dimensões em dois problemas a uma dimensão, funciona por redução do problema, utilizando ainda níveis de aspiração para decidir da aceitação ou não das soluções obtidas.

Christofides e Whitlock, 1977 [9]

"An Algorithm for Two-Dimensional Cutting Problems"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Maximizar o valor dos rectângulos cortados;

Restrições: Cortes em guilhotina;
Número de rectângulos restrito.

Neste artigo é apresentado um algoritmo que gera o padrão de corte óptimo, dado um conjunto de pedidos e uma folha de *stock*. É de realçar que este foi o primeiro algoritmo que surgiu para resolver o problema dos cortes rectangulares, ortogonais e com o número de rectângulos restrito, sem qualquer restrição adicional para além dos cortes serem em guilhotina. Todas as soluções conhecidas até à data consideravam o problema não restrito, ou então com restrições no número de fases de corte (problema obviamente mais simples de resolver).

Este algoritmo baseia-se numa pesquisa em árvore. A árvore de pesquisa é obtida considerando exaustivamente os vários cortes que é possível fazer na folha de *stock* e nas sub-folhas sucessivamente geradas pelos cortes efectuados. O tamanho da árvore é limitado derivando e impondo condições necessárias à optimalidade da solução correntemente em construção. Para tal, cada nó é avaliado:

- em termos do valor a obter, através da resolução do problema não restrito pelo procedimento de Programação Dinâmica de Gilmore e Gomory, que permite determinar o valor máximo que ainda é possível obter das áreas não cortadas;
- em termos do valor já obtido, através de uma rotina de transportes que vai determinar a afectação óptima dos pedidos aos rectângulos que já não vão sofrer mais nenhum corte, obtendo assim o valor inerente à parte da solução já construída.

A soma destes dois valores é comparada com o valor da melhor solução já obtida, só prosseguindo a pesquisa da solução testada se a soma for maior.

O algoritmo considera ainda os efeitos da simetria e da ordem pela qual os cortes são feitos, de forma a evitar a geração de padrões aparentemente diferentes, mas intrinsecamente iguais, ou seja, que cortam os mesmos rectângulos. Frequentemente citada na literatura, é a definição de padrões de corte normais que os autores fazem. Um padrão de corte normal é um padrão que só apresenta cortes em posições pré-definidas, e que correspondem a combinações lineares dos comprimentos ou larguras dos rectângulos pedidos. Desta forma o desperdício surge sempre encostado aos lados do padrão, característica pela qual se identifica um padrão normal.

Madsen, 1979 [27]

"Glass Cutting In a Small Firm"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimizar o desperdício;

Restrições: Cortes em guilhotina;
Cortes em duas fases.

É apresentada uma técnica que se baseia num duplo procedimento com funções mochila a uma dimensão, portanto, segundo a classificação apresentada em 4.1, uma heurística de redução do problema. Em primeiro lugar os rectângulos pedidos são agrupados de forma a formarem secções, e em seguida essas secções são combinadas de forma a formarem padrões a duas

dimensões, segundo os quais as folhas de *stock* serão posteriormente cortadas. O algoritmo de Gilmore e Gomory foi o utilizado para calcular as funções mochila.

Baker, Coffman e Rivest, 1980 [2]

"Orthogonal Packings in Two Dimensions"

Problema tratado: Uni-Padrão, 1 dimensão e 1/2 (rectângulo aberto no topo e sem fim).

Objectivo: Minimizar a altura total do empacotamento;

Restrições: Número de rectângulos restrito;
Orientação dos rectângulos fixa.

Neste artigo é estudado o desempenho das heurísticas de empacotamento tipo *Bottom-Up Left-Justified* (BL). Estas heurísticas caracterizam-se por colocarem um rectângulo o mais abaixo possível e, dentro do mesmo nível, o mais à esquerda possível. São ainda estabelecidos limites para os piores casos e estudada a influência da ordem pela qual os rectângulos são colocados, na eficiência das heurísticas. Conclui-se que se obtêm os melhores resultados quando a lista de rectângulos a colocar está ordenada por ordem decrescente de comprimentos.

Coffman, Garey, Johnson e Tarjan, 1980 [10]

"Performance Bounds for Level-Oriented Two-Dimensional Packing Algorithms"

Problema tratado: Uni-Padrão, 1 dimensão e 1/2 (rectângulo aberto no topo e sem fim).

Objectivo: Minimizar a altura total do empacotamento;

Restrições: Número de rectângulos restrito;
Orientação dos rectângulos fixa.

Neste artigo são consideradas três heurísticas de empacotamento: *Next-Fit Decreasing-Height* (NFDH), *First-Fit Decreasing-Height* (FFDH) e *Split-Fit* (SF).

Quer a NFDH, quer a FFDH assumem a lista de rectângulos ordenada por alturas não crescentes. Ambas as heurísticas fazem a colocação dos rectângulos por níveis: o primeiro nível corresponde ao fundo do rectângulo, enquanto os níveis seguintes correspondem a linhas horizontais que assentam no primeiro rectângulo (e logo o mais alto) do nível anterior. As duas heurísticas diferenciam-se pela estratégia de colocação: enquanto a NFDH vai colocando os rectângulos num dado nível até que um rectângulo lá não caiba, abrindo então novo nível e recomeçando aí o processo de colocação, a heurística FFDH tenta sempre colocar os rectângulos no nível mais abaixo possível. Assim, a FFDH só abrirá um novo nível se nenhum dos níveis já abertos puder acomodar o rectângulo que se pretende colocar.

A heurística SF divide os pedidos em duas listas: uma com os rectângulos com comprimento superior a uma certa quantidade, outra com os restantes. Em seguida, coloca os rectângulos mais compridos, usando a heurística FFDH, e rearranja-os de modo a criar uma zona livre onde possa colocar os rectângulos mais curtos, novamente através da FFDH.

Para as três heurísticas são estabelecidos limites superiores assintóticos para a sua eficiência, limites esses que relacionam a altura do empacotamento produzido pela heurística, com a altura do empacotamento óptimo: $A(L) \leq \beta \text{ OPT}(L) + \partial$. Os resultados obtidos foram os seguintes:

$$A(\text{NFDH}) \leq 2 \text{ OPT} + 1$$

$$A(\text{FFDH}) \leq 1.7 \text{ OPT} + 1$$

$$A(\text{SF}) \leq 1.5 \text{ OPT} + 1$$

Os autores afirmam ainda que estas heurísticas, em termos de limites superiores para a eficiência, são melhores que as heurísticas BL, cujo melhor valor assintótico é de 2 OPT .

Madsen, 1980 [28]

"A Cutting Sequencing Algorithm"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimizar o desperdício;

Sequenciar os padrões.

Restrições: Cortes em guilhotina;

Cortes em duas fases.

O problema posto ao autor deste artigo, o de minimizar o desperdício e simultâneamente fazer com certos pedidos sejam satisfeitos dentro de certo período de tempo, ou seja, dentro do mesmo padrão ou em padrões temporalmente adjacentes; foi por este resolvido formulando o problema em duas fases:

- numa primeira fase são gerados padrões com desperdício mínimo, ou seja, apenas com o objectivo de minimizar o desperdício;
- na segunda fase é determinada a ordem pela qual os padrões devem ser executados, ou seja, é resolvido um problema de sequenciamento.

Para a primeira fase o autor opta pela técnica já descrita em [27]. Quanto à segunda fase, são apresentadas várias técnicas de sequenciamento possíveis, tendo o autor optado pela aplicação do "método da redução da largura de banda", apresentando os resultados da sua aplicação ao problema concreto que pretendia resolver.

O "método da redução da largura de banda" assenta na construção de uma matriz simétrica binária, onde $a_{ij}=1$ significa que o padrão i e o padrão j têm pelo menos um pedido em comum, ou seja, cortam rectângulos que se pretende que saiam da linha de corte temporalmente próximos. É assumido que a ordem das linhas e das colunas (padrões) corresponde à sequência pela qual os padrões serão executados. O método consiste então em trocar as linhas e as colunas de forma a que a largura de banda da matriz seja mínima, isto é, de forma a que a matriz se concentre o mais possível em torno da sua diagonal principal.

De uma forma global, esta técnica insere-se na classe das heurísticas de redução do problema, uma vez que os seus dois objectivos são tratados separadamente (ver 4.1).

Tokuyama e Ueno, 1981 [32]

"The Cutting Stock Problem in the Iron and Steel Industries"

Problema tratado: Multi-Padrão, 2 dimensões.

- Objectivo:*
- Minimizar o desperdício;
 - Minimizar o número de sobras que serão guardadas para reaproveitamento;
 - Maximizar a eficiência das operações de corte;
 - Minimizar o número de chapas a cortar.
- Restrições:*
- Geométricas: - cortes em guilhotina;
 - Tecnológicas: - o bicos de corte transversal são muito menos eficiente que os bicos de corte longitudinal;
 - o processo de corte longitudinal só admite uma mudança na posição dos bicos de corte.

Neste artigo é apresentada uma solução heurística para um problema industrial concreto. A combinação de vários objectivos (alguns impostos pelas restrições do processo de fabrico) é feita por atribuição de pesos a cada um desses objectivos. Esses pesos variam conforme as necessidades do momento, embora seja sempre atribuída primordial importância à minimização do desperdício total e à eficiência do processo de corte.

A heurística propriamente dita, resolve o problema multi-padrão decompondo-o numa série de problemas uni-padrão. Para cada um destes problemas, agrupa os pedidos por larguras iguais, gerando em seguida, dentro de cada agrupamento, padrões de dimensão 1. Em seguida esses padrões são alocados transversalmente, e novamente por um algoritmo para problemas a uma dimensão. Para conseguir padrões de acordo com as restrições descritas, são estabelecidos graus

de preferência para os padrões a uma dimensão gerados, de forma a que sejam preferencialmente alocados com outros com as mesmas características, obtendo-se assim padrões mais homogêneos.

Esta heurística, para além de efectuar uma redução do problema ao tratar o problema a duas dimensões como dois problemas a uma dimensão, usa duplamente uma estratégia de valorização heurística (segundo definição apresentada em 4.1).

Baker, Brown e Katseff, 1981 [3]

"A 5/4 Algorithm for Two-Dimensional Packing"

Problema tratado: Uni-Padrão, 1 dimensão e 1/2 (rectângulo aberto no topo e sem fim).

Objectivo: Minimizar a altura total do empacotamento;

Restrições: Número de rectângulos restrito;
Orientação dos rectângulos fixa.

Neste artigo é apresentada uma heurística de empacotamento, que no pior dos casos apresenta uma solução cuja altura é igual a 5/4 da altura da solução óptima - a heurística *Up-Down* (UD).

Esta heurística funciona do mesmo modo que a NFDH, se todas os rectângulos tiverem um comprimento não superior a 1/5 da largura total do rectângulo onde se está a fazer o empacotamento. Se tal não acontecer, a heurística UD mistura várias estratégias de empacotamento, de modo a obter eficiências melhoradas para os casos extremos.

A heurística divide a região de empacotamento em 5 partes, à medida que vai empacotando os rectângulos. Para $1 \leq i \leq 4$ preenche a região R_i usando a heurística BL para colocar os rectângulos com comprimento entre $1/i+1$ e $1/i$ na região R_i . Como a heurística BL coloca cada rectângulo, por ordem não crescente de comprimentos, o mais abaixo e à esquerda possível, deixa um espaço livre de comprimento crescente do fundo para o topo, do lado direito da região. Tirando partido

desta configuração, a heurística UD coloca peças nessa coluna, de cima para baixo e por ordem crescente de comprimentos.

Quando todos os rectângulos com comprimento superior a $1/5$ tiverem sido colocados nas quatro regiões, segundo a estratégia acima descrita, os restantes rectângulos são colocados entre aqueles que foram colocados pela BL e a coluna complementar, através de uma generalização da heurística NFDH. Finalmente, os rectângulos que eventualmente ainda sobrem, são colocados no topo do empacotamento pela heurística NFDH.

Israni e Sanders, 1982 [25]

"Two-Dimensional Cutting Stock Problem Research: a Review and a New Rectangular Layout Algorithm"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Minimizar o desperdício.

Restrições: Número de rectângulos restrito.

Neste trabalho, os autores apresentam duas heurísticas para o problema dos cortes a duas dimensões, baseadas na heurística de empacotamento *First-Fit-Decreasing*.

Na primeira heurística, a lista de pedidos é ordenada por ordem decrescente de comprimentos, sendo depois colocada ao longo de um semi-perímetro da folha de *stock*. A parte da folha não ocupada é "arredondada" para o maior rectângulo que nela se inscreve e este rectângulo considerado para colocação dos pedidos restantes. O processo prossegue iterativamente até que todos os pedidos estejam satisfeitos, ou a área livre seja menor que o menor dos pedidos.

A segunda heurística funciona da mesma maneira, apenas diferindo na ordenação inicial dos pedidos, que é feita por ordem decrescente de larguras.

Wang, 1983 [34]

"Two Algorithms for Constrained Two-Dimensional Cutting Stock Problems"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Minimizar o desperdício.

Restrições: Cortes em guilhotina;
Orientação dos rectângulos fixa;
Número de rectângulos restrito.

Neste artigo são apresentadas duas versões de um algoritmo combinatório para o problema de cortes a duas dimensões. Os padrões de corte são gerados por combinando sucessivamente os rectângulos pedidos uns com os outros, e com combinações já feitas. Para limitar o número de combinações, os algoritmos utilizam níveis de aspiração, que essas combinações devem satisfazer para que continuem a ser consideradas no processo combinatório. Esses níveis de aspiração baseiam-se em parâmetros fornecidos pelo utilizador e traduzem percentagens de desperdício admissível, num caso percentagem da área já ocupada, no outro percentagem da área total da chapa de *stock*. O autor apresenta ainda alguns teoremas, que permitem estabelecer condições suficientes de optimalidade para os padrões gerados com um determinado valor dos parâmetros, assim como medir a distância à solução óptima dos padrões encontrados.

Biró e Boros, 1984 [8]

"Network Flows and Non-Guillotine Cutting Patterns"

Problema tratado: Uni-Padrão, 1 dimensão e meia.

Objectivo: Minimizar a altura total do empacotamento.

Restrições: Orientação dos rectângulos fixa;
Número de rectângulos restrito.

Neste artigo é feita uma caracterização, em termos da teoria dos grafos e fluxos em redes, de um padrão de corte com as restrições indicadas. É de realçar que não é considerada a restrição

dos cortes terem de ser do tipo guilhotina. Baseado nessa caracterização, os autores do artigo derivam uma heurística para a geração de padrões de corte.

Harrison, 1984 [22]

"The Two-Dimensional Cutting Problem of the Furniture Manufacturer"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimizar o desperdício;
Manter dentro de certos limites o número de padrões diferentes;
Sequenciar os padrões.

Restrições: Cortes em guilhotina;
Orientação dos rectângulos fixa;
Número de rectângulos restrito.

É apresentado um estudo do modo como os operários, treinados na geração manual de padrões de corte, procedem para gerar padrões de corte, de modo a obter um equilíbrio entre os vários objectivos a atingir. Em seguida é apresentada uma heurística que procura mimetizar esse procedimento manual. A heurística caracteriza-se por:

- criar, interactivamente, e manter uma biblioteca de "bons" padrões, onde vai em primeiro lugar procurar a solução para os problemas;
- gerar novos padrões através da criação de riscas e posterior selecção e colocação dessas riscas de modo a formar um padrão a duas dimensões;
- a selecção dos padrões de corte ser feita com base em pesos que são atribuídos pelo utilizador, a cada um dos objectivos do problema.

Esta heurística recorre simultâneamente a três técnicas diferentes (segundo a classificação proposta em 4.1). Por um lado, usa uma estratégia de amostragem ao procurar a solução numa biblioteca de padrões anteriormente gerados. Em seguida, ao criar novos padrões, faz uma

redução do problema. Finalmente, combina os diferentes objectivos do problema através de uma valorização heurística de cada um deles.

Beasley, 1985 [4]

"Algorithms for Unconstrained Two-Dimensional Guillotine Cutting"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Maximizar o valor dos rectângulos cortados.

Restrições: Orientação dos rectângulos fixa;
Cortes em guilhotina.

Neste artigo Beasley discute os algoritmos de Gilmore e Gomory [19], apresenta alguns melhoramentos e inclusivé detecta um erro num desses algoritmos. Começando por considerar o algoritmo de programação dinâmica para o problema do corte faseado (cortes em x fases), Beasley apresenta uma outra recursão e mostra que o algoritmo de Gilmore e Gomory comporta incorrecções na medida em que não considera todas as orientações possíveis para o primeiro corte. A partir da consideração dos padrões normais, introduzidos por Christofides [9], o autor diminui o conjunto dos pontos de corte admissíveis, aumentando assim a eficiência do seu algoritmo.

Seguidamente, é considerado o caso de não haver restrições no número de fases em que os cortes são feitos (padrões em n -fases). Apontando as desvantagens do algoritmo que Herz [23] apresentou para este mesmo problema, Beasley pega no algoritmo de programação dinâmica de Gilmore e Gomory [19], implementa-o, introduz-lhe alguns melhoramentos computacionais e apresenta alguns resultados computacionais. O autor apresenta ainda um procedimento heurístico, baseado neste algoritmo, com particular aplicação em problemas grandes, onde o algoritmo original de Gilmore e Gomory se torna bastante ineficiente.

Beasley, 1985 [5]

"An Exact Two-Dimensional Non-Guillotine Cutting Tree Search Procedure"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Maximizar o valor dos rectângulos cortados.

Restrições: Orientação dos rectângulos fixa;
Número de rectângulos entre dois limites.

O autor apresenta um procedimento de pesquisa em árvore, cuja expansão é controlada através da introdução de limites. Estes limites são obtidos a partir da relaxação Lagrangiana da formulação do problema como programação inteira binária, sendo ainda o limites otimizados através da optimização por subgradiente.

A técnica apresentada permite a inclusão de defeitos nas chapas de *stock*. O tratamento de chapas de *stock* múltiplas (problema multi-padrão) é feito por junção lado a lado das chapas individuais, tratando-as depois como uma só chapa, embora com algumas restrições adicionais de modo a que nenhum rectângulo seja colocado na união de duas chapas.

Toczyłowsky, 1986 [31]

"On Aggregation in a Two-Dimensional Cutting Stock Problem"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimizar o desperdício;
Minimizar as sobreproduções.

Restrições: Número de rectângulos limitado inferiormente.

No problema apresentado neste artigo, embora existam quantidades pedidas bem definidas, são permitidas sobreproduções. Contudo a essas sobreproduções é associado um custo de armazenagem, que somado ao desperdício realmente gerado, deve ser minimizado. Por outro lado,

os pedidos podem ser satisfeitos a partir de rectângulos cortados expressamente para esse cliente, ou a partir de sobreproduções de processos de corte anteriores, que se encontrem em armazém.

A resolução do problema baseia-se numa formulação mais compacta que a formulação clássica do problema multi-padrão, resultando na agregação dos conjuntos de rectângulos e padrões de corte. O método permite explorar as propriedades estruturais do problema para criar um problema de Programação Linear Inteira equivalente mas mais conciso. Este modelo contém consideravelmente menos variáveis e restrições.

Durante o processo de agregação, rectângulos e padrões similares são agrupados em classes de equivalência. Cada classe vai corresponder a uma só variável ou restrição no modelo agregado. Dois rectângulos são considerados similares se tiverem as mesmas dimensões. Dois padrões são similares se substituídos um pelo outro derem ainda origem a um padrão admissível.

A eficiência do método depende fortemente da estrutura do problema: quanto maior for o número de rectângulos similares, maiores as vantagens do modelo agregado face ao modelo de Programação Linear clássico. O autor afirma que o método é promissor para a resolução de problemas de tamanho moderado.

Berkey e Wang, 1987 [7]

"Two-Dimensional Finite Bin-Packing Algorithms"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimizar o número de chapas de *stock* utilizadas;

Restrições: Número de rectângulos restrito.

Neste artigo são apresentadas heurísticas adaptadas de heurísticas usadas na resolução de problemas de empacotamento (rectângulo aberto no topo), para o caso de rectângulos fechados. As

heurísticas adaptadas são do tipo *First-Fit*, *Next-Fit* e *Best-Fit*, portanto todas elas com colocação por níveis. As heurísticas apresentadas são as seguintes:

- *Finite Next-Fit* (FNF)
- *Finite First-Fit* (FFF)
- *Finite Best-Strip* (FBS)
- *Finite Bottom-Left* (FBL)

As heurísticas FNF, FFF, FBL funcionam basicamente como as suas congêneres para o problema do rectângulo de altura infinita. A diferença acontece quando um rectângulo finito acaba, não se podendo portanto abrir mais níveis, o novo nível é aberto num novo rectângulo. A heurística FBS tem uma abordagem ligeiramente diferente. Em primeiro lugar os rectângulos são empacotados num rectângulo de altura infinita, sendo depois esse empacotamento dividido em blocos, de forma a caberem nos rectângulos finitos. Esta última afectação dos blocos aos rectângulos finitos é feita através de um heurística tipo *Best-Fit*.

Farley, 1987 [15]

"Practical Adaptations of the Gilmore-Gomory Approach to Cutting Stock Problems"

Problema tratado: Multi-Padrão, 2 dimensões.

Objectivo: Minimizar o número de chapas de *stock* utilizadas;

Restrições: Cortes em guilhotina;
Cortes em duas ou três fases.

Neste artigo, Farley aborda técnica de geração atrasada de colunas, proposta por Gilmore e Gomory [16], para problemas de cortes a duas dimensões. São considerados padrões guilhotináveis e em duas ou três fases. Contudo, conforme o autor afirma, muitos dos conceitos aqui introduzidos podem ser facilmente extendidos a outras dimensões ou outros tipos de padrões.

Farley apresenta a abordagem de Gilmore e Gomory, assim como o algoritmo de pesquisa lexicográfica (destes mesmos autores) utilizado na resolução do problema auxiliar (um problema mochila) que esta abordagem supõe. O algoritmo de pesquisa lexicográfica é um algoritmo para problemas a uma dimensão, sendo utilizado na resolução de dois problemas sucessivos a uma dimensão que darão origem a um padrão a duas dimensões em duas ou três fases. Em seguida são apresentados alguns melhoramentos à abordagem original, nomeadamente na:

- geração do conjunto inicial de padrões;
- consideração de soluções múltiplas para o problema mochila;
- valorização explícita de sub e sobreproduções;
- consideração de um valor para o desperdício.

A técnica da programação linear com geração atrasada de colunas, baseando-se na filosofia de gerar uma coluna do problema PL só e quando ela for necessária para entrar na base, supõe um conjunto inicial de colunas para que a técnica possa arrancar. Farley apresenta várias hipóteses para a constituição desse conjunto, desde cada padrão conter só uma peça até gerar padrões de desperdício mínimo. Essas hipóteses foram implementadas, testadas e os resultados computacionais apresentados. Outra questão abordada tem a ver com a resolução do problema mochila. Habitualmente considera-se a solução óptima desse problema, gerando-se o respectivo padrão e introduzindo-o na base. Como o critério de entrada de variáveis na base é o da minimização dos custos marginais, critério de algum modo local e de carácter relativo, poderia ser vantajoso considerar mais que uma solução do problema mochila e introduzir mais que um padrão de cada vez no problema PL. Farley desenvolve esta ideia e apresenta resultados da sua implementação. Farley ainda modeliza as situações em que a existência de sub e/ou sobreproduções, sem ser proibida, deve ser penalizada, assim como aqueles casos em que o desperdício pode ser reaproveitado em processos de corte posteriores, tendo portanto associado um valor, eventualmente dependente da sua forma ou tamanho. Finalmente, Farley discute as várias técnicas de obtenção de uma solução inteira, uma vez que o modelo utilizado relaxa essa restrição.

Vasko, 1989 [33]

"A Computational Improvement to Wang's Two-Dimensional Cutting Stock Algorithm"

Problema tratado: Uni-Padrão, 2 dimensões.

Objectivo: Minimizar o desperdício.

Restrições: Cortes em guilhotina;
Orientação dos rectângulos fixa;
Número de rectângulos restrito.

Neste artigo Vasko apresenta algumas modificações que introduziu no algoritmo de Wang [34], de modo a aumentar a sua eficiência computacional. É ainda apresentada uma heurística, a SPAM, para o problema similar a este mas com os cortes em duas fases.

Vasko utiliza a heurística SPAM para gerar uma solução inicial para o algoritmo de Wang, e assim determinar um valor inicial para o parâmetro β , percentagem de desperdício interno admissível. Em seguida introduz algumas alterações no algoritmo original na parte da construção das soluções parciais. Assim, definindo o conceito de uma solução parcial ser verticalmente (horizontalmente) completa, isto é, não é possível colocar em cima (ao lado) mais nenhum rectângulo sem que se excedam as dimensões da chapa de *stock*, Vasko exclui das construções verticais (horizontais) as soluções parciais verticalmente (horizontalmente) completas. Quando uma solução parcial se revela quer verticalmente, quer horizontalmente, completa então o seu desperdício total é calculado e o valor de β actualizado, caso esse desperdício seja inferior ao da melhor solução corrente. Vasko obtém assim um algoritmo que ajusta dinamicamente o valor do parâmetro β . Os resultados computacionais apresentados revelam a vantagem da utilização destes melhoramentos no algoritmo de Wang.

PARTE II

TÉCNICAS UNI-PADRÃO

Capítulo 5

Algoritmos uni-padrão para uma abordagem optimizante do Problema dos Cortes rectangulares.

Neste capítulo apresenta-se uma descrição de três algoritmos para o problema dos cortes rectangulares a duas dimensões: o algoritmo de Christofides e Whitlock, o algoritmo de Wang e o algoritmo de Wang modificado. Este algoritmo modificado, que foi desenvolvido por nós, baseia-se na abordagem do algoritmo de Wang [34].

Usando a classificação proposta no capítulo 4 desta dissertação, para as técnicas de resolução de problemas de cortes, estes três algoritmos inserem-se na classe das técnicas uni-padrão. Ainda dentro da classificação proposta, estas técnicas são de facto do tipo algorítmico, uma vez que, considerando a formulação do problema como uni-padrão, encontram a solução óptima.

Recorrendo agora à classificação apresentada no capítulo 3, os problemas que estes algoritmos resolvem caracterizam-se por considerarem padrões de corte ortogonais, em guilhotina e sem rotações dos rectângulos. O número de rectângulos de cada tipo, utilizados num padrão de corte, não pode exceder as quantidades pedidas. Quanto ao objectivo, para o algoritmo de Christofides e Whitlock é o de maximizar o valor extraído da folha de *stock*, enquanto para o algoritmo de Wang e para o algoritmo modificado é o de minimizar os desperdícios. Recorda-se que, quando os valores atribuídos aos rectângulos são proporcionais às respectivas áreas, estes objectivos são equivalentes.

5.1 - Formulação e modelos.

A solução do problema descrito, e para o qual se pretende apresentar uma formulação, é um padrão de corte, ou seja, uma combinação de rectângulos, retirados do conjunto de rectângulos pedidos, que pode ser cortada de uma folha de *stock* e onde o número de vezes que um rectângulo pode aparecer é limitado. Os algoritmos que serão apresentados geram padrões normalizados, no sentido apresentado por Christofides e Whitlock [9], sendo apenas considerados padrões deste tipo.

Um padrão de corte estará então completamente definido pelo seu comprimento, largura e pelo número de vezes que cada rectângulo aparece, juntamente com a respectiva localização no padrão de corte.

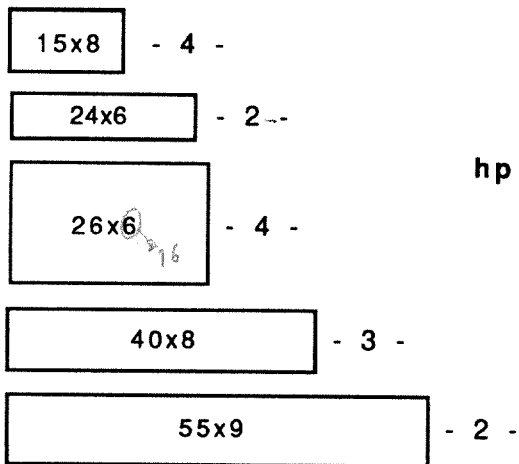
Considere-se então a seguinte notação:

- w - comprimento da folha de *stock*;
- h - largura da folha de *stock*;
- m - número de rectângulos diferentes a cortar;
- w_p - comprimento do padrão de corte;
- h_p - largura do padrão de corte;
- w_i - comprimento do rectângulo i ;
- h_i - largura do rectângulo i ;
- b_i - número máximo de vezes que o rectângulo i pode ser usado;
- x_i - número de vezes que o rectângulo i é efectivamente usado num padrão de corte.

Um padrão de corte terá então que satisfazer as condições $w_p \leq w$ e $h_p \leq h$. A figura 5.1 apresenta um padrão de corte, com as restrições consideradas, e ilustra parte da notação definida.

Diferentes peças a cortar:

$w_i \times h_i$ - b_i -



Um padrão de corte

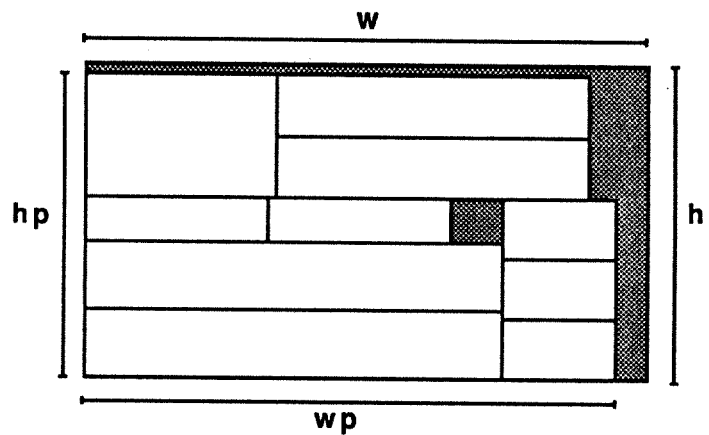


Figura 5.1 - Um exemplo de um padrão de corte.

A solução ótima deste problema é um padrão de corte, como foi acima definido, que minimiza o desperdício, isto é, que é a solução do seguinte modelo:

$$\text{minimizar} \quad wh - \sum_{i=1}^m x_i w_i h_i$$

$$\text{sujeito a:} \quad 0 \leq x_i \leq b_i, \quad 1 \leq i \leq m$$

$$x_i \text{ inteiro}$$

Se existirem valores atribuídos aos rectângulos, então a função objectivo vem:

$$\text{maximizar} \quad \sum_{i=1}^m x_i v_i$$

onde v_i é o valor atribuído ao rectângulo i .

5.2 - O algoritmo de Christofides e Whitlock.

Introdução

Christofides e Whitlock [9] apresentaram um algoritmo de pesquisa em árvore para a resolução do problema dos cortes rectangulares a duas dimensões, em que há restrições no número máximo de vezes que cada rectângulo pode aparecer num padrão de corte, e em que os cortes são em guilhotina. O algoritmo limita o tamanho da árvore de pesquisa impondo ao padrão de corte condições necessárias de optimalidade. Estas condições baseiam-se na avaliação de cada nó da árvore, através da resolução de um problema sem restrições, por um procedimento de programação dinâmica, e através da resolução de um problema de transportes. Consegue-se assim calcular limites superiores para o valor do padrão de corte, ainda durante a expansão e pesquisa da árvore que lhe dará origem.

A formulação do problema utilizada por Christofides, tem a particularidade de admitir para os rectângulos valores não proporcionais às respectivas áreas, sendo o objectivo do problema, não a minimização do desperdício, mas sim a maximização do valor que se extrai desse mesmo padrão de corte.

Geração dos padrões de corte

A geração de todos os padrões de corte admissíveis, pode ser feita desenvolvendo uma árvore em que os ramos representam cortes num rectângulo e os nós o conjunto dos rectângulos já gerados (por corte de outros rectângulos). Esse conjunto será constituído por todos os rectângulos do nó anterior, excepto aquele que foi cortado que é substituído pelos dois novos rectângulos gerados. Assim, os ramos que saem da raiz da árvore representam todos os cortes possíveis na folha de *stock*.

Para que se possa gerar exhaustivamente todos os padrões de corte, é necessário incluir entre os cortes possíveis o corte-0. Este tipo de corte corresponde a não fazer corte nenhum, isto é, a deixar o rectângulo intacto.

Em cada nó vai-se sucessivamente escolhendo um rectângulo da lista de rectângulos que representa o nó, e ramifica-se a partir desse nó fazendo todos os cortes possíveis no rectângulo escolhido, até que todos os rectângulos tenham sido escolhidos.

Durante este processo de ramificação, procura-se evitar a duplicação dos padrões de corte devido ao efeito da simetria. Dado um rectângulo de dimensões (w,h) , se se fizer um corte nesse rectângulo em $x=a$, obtêm-se dois novos rectângulos de dimensões (a,h) e $(w-a,h)$. Claramente, estes mesmos dois subrectângulos poderiam ter sido produzidos por um corte em $x=w-a$. Esta situação é evitada restringindo os cortes segundo x ao intervalo de 1 ao maior inteiro que está contido em $w/2$, e segundo y ao intervalo de 1 ao maior inteiro que está contido em $h/2$, ou seja:

$$1 \leq x \leq \text{int}(w/2)$$

$$1 \leq y \leq \text{int}(h/2)$$

Contudo, a duplicação dos padrões de corte também pode acontecer devido à ordem pela qual os cortes são produzidos. Corte-se um rectângulo (w,h) segundo $x=a$. Obtêm-se os dois subrectângulos (a,h) e $(w-a,h)$. Suponha-se agora que o rectângulo $(w-a,h)$ é cortado segundo $x=b$. Fica-se então com três rectângulos de dimensões (a,h) , (b,h) e $(w-a-b,h)$. Estes mesmos rectângulos poderiam ter sido obtidos fazendo primeiro um corte segundo $x=b$ em (w,h) e depois segundo $x=a$ em $(w-b,h)$. Este tipo de duplicação pode ser evitada introduzindo uma ordenação na sequência dos cortes. Assim, se um rectângulo fôr cortado segundo $x=\alpha$, todos os subsquentes cortes segundo x nos subrectângulos resultantes devem ser feitos para $x \geq \alpha$. Esta restrição, juntamente com aquela devida aos efeitos de simetria, conduz-nos às seguintes limitações:

- se um rectângulo (w,h) fôr cortado segundo $x=\alpha$, então no maior dos dois subrectângulos resultantes, $(w-\alpha,h)$, só poderão ser feitos cortes segundo x tais que

$$\alpha \leq x < \text{int}((w - \alpha)/2)$$

enquanto no subrectângulo (α, h) não se podem fazer mais cortes segundo x . Condições semelhantes podem ser derivadas para os cortes segundo y .

Christofides e Whitlock introduzem ainda o conceito de padrão normalizado. Assim, para que os padrões gerados venham normalizados, introduz-se uma restrição adicional aos cortes segundo x e segundo y , isto é, limita-se os cortes segundo x aos valores x_0 tais que existe uma combinação linear de comprimentos w_i (dos rectângulos pedidos) tal que:

$$h_i \leq h$$

$$\sum_i l_i w_i = x$$

e limita-se os cortes segundo y aos valores de y tais que existe uma combinação linear de larguras h_i tal que:

$$w_i \leq w$$

$$\sum_i l_i h_i = y$$

Seja S^h o conjunto de valores de x segundo os quais pode ser feito um corte segundo x no rectângulo (w, h) , dado o conjunto dos rectângulos a cortar (w_i, h_i) . Assuma-se ainda que os m rectângulos estão ordenados por ordem não crescente de valores de h_i . Define-se então a função $f_r(x)$ do seguinte modo:

- se existe um ou mais vectores V ($V \in \mathbb{N}^n$) tais que

$$x = \sum_{i=1}^r V_i w_i, \quad 0 \leq V_i \leq b_i$$

então faz-se

$$f_r(x) = h_{i^*}$$

onde

$$i^* = \min_V [\max \{ i \mid V_i \neq 0 \}]$$

- se não existe tal vector V então faz-se

$$f_r(x) = \infty$$

$f_r(x)$ vai então conter, para cada valor de x , a largura do rectângulo mais largo que faz parte da combinação linear de rectângulos mais favorável (aquela cuja largura máxima é menor), que permite atingir x - isto quando apenas se consideram rectângulos dos primeiros r tipos. Não existindo nenhuma combinação essa largura máxima será igual a ∞ . Pertencerão ao conjunto S^h todos os valores de x para os quais $f_m(x)$ (usam-se todos os tipos de rectângulos) seja menor ou igual que h . Note-se que uma vez gerada a tabela $f_m(x)$, os conjuntos S^h podem ser criados para qualquer h .

Computacionalmente, a tabela $f_r(x)$ pode ser gerada através da seguinte fórmula recursiva:

$$f_i(x) = \min \{ f_{i-1}(x) , \max_k \{ h_i , \min (f_{i-1}(x - kw_i)) \} \}$$

onde:

$$1 \leq i \leq m$$

$$0 \leq x \leq w$$

$$1 \leq k \leq \min \{ b_i , \text{int}(x/w_i) \} , k \text{ inteiro e } x \geq w_i$$

$$f_i(x) = f_{i-1}(x) \text{ se } x < w_i$$

$$f_0(x) = \infty \text{ para todo } x$$

Esta equação calcula a menor das máximas larguras dos rectângulos cujas combinações lineares de comprimentos atingem o valor x (usando todos os tipos até ao tipo i), como o menor valor entre essa mesma máxima largura, mas só usando rectângulos até ao tipo $i-1$, e a máxima largura que se obtem usando rectângulos do tipo i . Para o cálculo deste último valor, a equação decide quantos rectângulos do tipo i devem entrar na nova combinação de comprimentos, de novo para que a largura máxima seja a menor possível.

Este processo é repetido para os cortes segundo y , gerando de um modo similar os conjuntos T^w .

Procedimento de pesquisa

Definidos quais os valores de x e y segundo os quais se podem efectuar cortes, descreve-se agora o algoritmo que gera os padrões de corte propriamente ditos.

O estado da pesquisa, correspondente a um nó n , é descrito pela lista L dos rectângulos produzidos pela sequência de cortes correspondentes ao caminho entre a raiz da árvore e o nó n . Na lista L cada rectângulo é representado pelo quarteto de valores (w,h,x,y) , onde w e h são o comprimento e a largura do rectângulo, e x e y são inteiros que tomam os seguintes valores:

$$0 \leq x \leq \text{int}(w/2) + 1$$

$$0 \leq y \leq \text{int}(h/2) + 1$$

O significado - para o processo de pesquisa - dos valores de x e y é o seguinte:

- se $1 \leq x \leq \text{int}(w/2)$ então o próximo corte a ser considerado para o rectângulo (w,h) é um corte segundo x na posição x ;
- se $x = \text{int}(w/2) + 1$ e $1 \leq y \leq \text{int}(h/2)$ então o próximo corte é um corte segundo y na posição y ;
- se $x = \text{int}(w/2) + 1$ e $y = \text{int}(h/2) + 1$ então todos os cortes admissíveis, segundo x e segundo y , já foram feitos e o próximo corte terá que ser um corte-0;
- se $x = 0$ infere-se que já foi feito um corte-0 no rectângulo (w,h) , pelo que não deve tornar a ser cortado.

O algoritmo de pesquisa pode então ser descrito do seguinte modo:

$L := \{(w, h, 1, 1)\};$

enquanto não tiverem sido tentados todos os cortes **faz**

 escolhe um rectângulo (w_r, h_r, x, y) de L , tal que $x \neq 0$;

se existe um rectângulo nessas condições **então**

$L := L - \{(w_r, h_r, x, y)\};$

enquanto $x < \text{int}(w_r/2) + 1$ **faz**

se $x \in S^{h_r}$ **então**

$L := L \cup \{(w_r - x, h_r, x, 1), (x, h_r, x, 1)\};$

senão $x := x + 1$;

se $x = \text{int}(w_r/2) + 1$ **então**

enquanto $y < \text{int}(h_r/2) + 1$ **faz**

se $y \in T^{w_r}$ **então**

$L := L \cup \{(w_r, h_r - y, 1, y), (w_r, y, 1, y)\};$

se $y = \text{int}(h_r/2) + 1$ **então**

$L := L \cup \{(w_r, h_r, 0, y)\};$

senão

 vai buscar os últimos rectângulos cortados:

$L := L - \{(w_1, h_1, x_1, y_1), (w_2, h_2, x_2, y_2)\};$

se $h_1 = h_2$ **então** $L := L \cup \{(w_1 + w_2, h_1, x + 1, 1)\};$

se $w_1 = w_2$ **então** $L := L \cup \{(w_1, h_1 + h_2, 1, y + 1)\};$

Este procedimento de pesquisa é exaustivo, isto é, tenta todos os cortes possíveis em todos os rectângulos. Numa tentativa de acelerar este algoritmo, Christofides e Whitlock introduziram-lhe critérios para limitar a expansão de alguns nós.

Avaliação dos nós - limite superior para o valor de um padrão de corte

Seja n um nó da árvore de pesquisa. Seja L a lista dos rectângulos já cortados. Seja ainda H_0 o subconjunto de L que contém todos os rectângulos que já sofreram um corte-0. Estes rectângulos não serão pois mais cortados em nenhum dos nós abaixo do nó corrente. No padrão final, estes rectângulos irão ter algum dos rectângulos pedidos, r_l , encaixado dentro deles. Então, um limite superior para o valor obténível dos rectângulos pertencentes a H_0 pode ser conseguido a partir de uma afectação óptima destes rectângulos aos rectângulos pedidos. Para tal, o problema é formulado como um problema de transportes e resolvido por uma rotina adequada. A formulação para o problema é a seguinte:

$$\begin{aligned} \max z &= \sum_{k=1}^u \sum_{l=1}^m a_{lk} x_{lk} \\ \text{sujeito a: } &\sum_{l=1}^m x_{lk} \leq 1 \\ &\sum_{k=1}^u x_{lk} \leq b_l \\ &x_{lk} \geq 0 \end{aligned}$$

em que:

$[a_{lk}]$ - matriz com m linhas, correspondendo aos m tipos de rectângulos pedidos, e u colunas, correspondentes aos u rectângulos (w_k, h_k) de H_0 ;

$$\text{Matriz dos custos: } \begin{cases} a_{lk} = v_l & \text{se } w_l \leq w_k \text{ e } h_l \leq h_k \\ a_{lk} = -\infty & \text{se não} \end{cases}$$

Os rectângulos nos quais ainda não foram feitos cortes-0, são passíveis de futuros cortes, gerando outros rectângulos menores e assim sucessivamente. Contudo, embora ainda não se saiba como esses rectângulos vão ser divididos, é também possível estabelecer um limite superior para o valor a obter desses rectângulos. Esta estimativa pode ser obtida a partir da resolução de um problema não restrito, isto é, em que não há restrições ao número de rectângulos de cada tipo a cortar. De facto, a solução do problema restrito constitui um limite

superior para o valor da solução do mesmo problema, mas com restrições. Este problema não restrito pode ser eficientemente resolvido através do algoritmo de Gilmore e Gomory [19].

Então, o limite superior para o valor a extrair de um dado nó, vai ser a soma destes dois valores: o valor correspondente aos rectângulos que já sofreram cortes-0 e a estimativa do valor correspondente aos rectângulos ainda não completamente cortados.

Seja então Z^* o valor do limite superior para um dado nó. Seja ainda Z^m o valor da melhor solução encontrada até ao momento. Se $Z^* > Z^m$ e o número de vezes que cada rectângulo i é usado nesta solução não excede b_i , então encontrou-se uma solução melhor do que a que já existia, passando Z^m a valer Z^* . Por outro lado, se $Z^* \leq Z^m$ não vale a pena prosseguir a expansão do nó, pois o valor que, na melhor das hipóteses, se iria extrair desse nó é inferior ao valor da melhor solução já conhecida.

É de salientar que, ao contrário do problema de transportes que necessita de ser resolvido em cada nó, o problema de programação dinâmica é resolvido uma única vez. As posteriores avaliações dos nós reduzem-se a leituras de uma tabela gerada pela rotina de programação dinâmica.

Estratégias de ramificação

Até ao momento, foi deixado por explicitar o critério de escolha do rectângulo da lista L que em cada nó será cortado. Christofides e Whitlock apresentam três hipóteses para o critério de selecção do rectângulo a cortar:

- (i) Seleccionar o rectângulo com menor comprimento.
- (ii) Seleccionar o rectângulo para o qual o problema não restrito dá o maior valor.
- (iii) Seleccionar o rectângulo que torna o problema mais "não admissível". Esta estratégia procura obter rapidamente uma solução admissível para o problema. Em cada nó r'_i rectângulos do tipo i são afectados aos rectângulos cortados (pela rotina de transportes) e r''_i são usados pelas soluções dos problemas não restritos, que são resolvidos para os rectângulos não

completamente cortados. Seja $r_i = r'_i + r''_i$. Então, necessariamente existe algum i para o qual $r_i > b_i$, pois caso contrário esta seria uma solução admissível e a expansão do nó teria terminado. Seja ainda i^* o rectângulo para o qual $r_i - b_i$ é máximo. A ideia subjacente a esta estratégia de ramificação é escolher o rectângulo que na solução não restrita usa mais rectângulos do tipo i^* . Assim, espera-se que depois de esse rectângulo ser novamente cortado, menos rectângulos do tipo i^* sejam usados e se tenha dado um passo na direcção da admissibilidade da solução.

É de salientar que, enquanto as duas primeiras estratégias não implicam grandes custos, do ponto de vista computacional, a terceira implica algum cálculo adicional que pode pesar no tempo final de execução. Contudo, embora o tempo médio gasto com cada nó seja maior, espera-se que o tempo total seja menor, uma vez que menos nós serão expandidos. É esta a estratégia de ramificação que Christofides e Whitlock usaram nos testes computacionais que efectuaram.

5.3 - O algoritmo de Wang.

Introdução e definição dos conceitos envolvidos

O algoritmo de Wang pertence à classe dos algoritmos combinatórios. Ele gera os padrões de corte fazendo-os crescer a partir de combinações sucessivas dos rectângulos pedidos, uns com os outros. A estas combinações de rectângulos será dado o nome de soluções parciais. Uma solução parcial dará origem a um padrão de corte quando, não se podendo adicionar mais nenhum rectângulo sem que alguma das restrições do problema seja violada, essa solução é colocada sobre uma folha de *stock*.

Para evitar o crescimento explosivo do número de soluções parciais, inerente ao carácter combinatório do algoritmo, é definido um nível de aspiração que as soluções parciais devem satisfazer, sem o que serão rejeitadas. Para tal, são usados parâmetros de rejeição que estão

directamente ligados ao desperdício gerado pela solução parcial. Este desperdício total pode ser decomposto em duas parcelas:

- uma parcela corresponde à diferença entre a área do menor rectângulo que envolve a solução parcial e a área efectivamente ocupada pela solução - este desperdício será designado por desperdício interno;

- quando se coloca uma solução parcial na folha de *stock*, dando assim origem a um padrão de corte, gera-se mais um desperdício corresponde à diferença entre a área da folha de *stock* e a área do rectângulo envolvente da solução parcial associada ao padrão de corte - este desperdício tomará a designação de desperdício externo.

O desperdício total será a soma do desperdício interno com o desperdício externo. Estes conceitos estão ilustrados na figura 5.2.

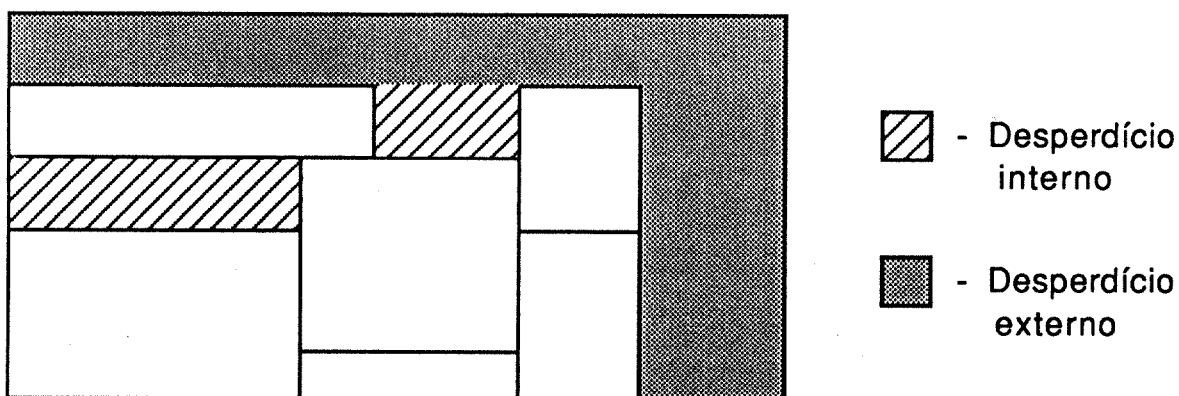


Figura 5.2 - Uma solução parcial e os dois tipos diferentes de desperdício.

A partir destes valores de desperdício, e dos valores do parâmetro de rejeição utilizados, são derivadas condições suficientes de optimalidade para o padrão de corte gerado nessas circunstâncias.

A técnica combinatória utilizada no algoritmo de Wang, usa os conceitos de construção vertical e horizontal de dois rectângulos. Define-se construção horizontal de dois rectângulos $A_1=p_1 \times q_1$ e $A_2=p_2 \times q_2$ como o rectângulo S_h de dimensões $\max(p_1, p_2) \times (q_1 + q_2)$ e que

contém A_1 e A_2 . Uma construção vertical de A_1 com A_2 será um rectângulo S_v com dimensões $(p_1+p_2) \times \max(q_1, q_2)$ e que contém A_1 e A_2 . Para que os padrões de corte venham normalizados, como definido por Christofides e Whitlock [9], todos os rectângulos são colocados o mais à esquerda e abaixo possível. Na figura 5.3 é apresentado um exemplo destas construções.

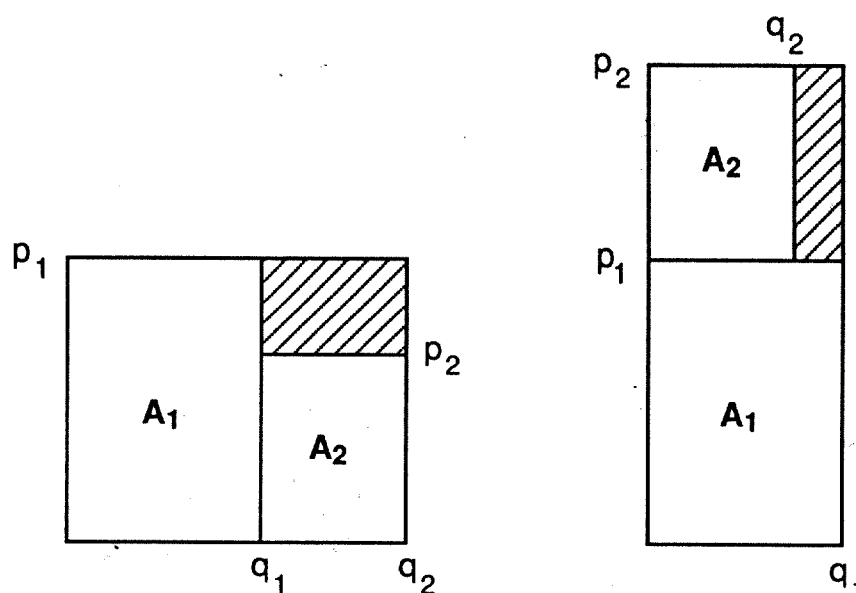


Figura 5.3 - Uma construção horizontal e vertical de A_1 e A_2

Wang apresenta duas versões do seu algoritmo, que diferem apenas no parâmetro de rejeição utilizado. Na primeira versão são rejeitadas todas as soluções parciais cujo desperdício interno exceda uma dada percentagem da área da folha de *stock*, enquanto na segunda versão essa percentagem se refere à área do rectângulo envolvente da solução parcial. Considerando a semelhança entre estas duas versões, e o facto de o algoritmo modificado - que será apresentado na próxima secção - assentar na primeira versão, a exposição do algoritmo de Wang será feita com base nessa versão.

Descrição do algoritmo de Wang

Como já foi apontado, o algoritmo de Wang faz sucessivas construções verticais e horizontais com as soluções parciais que vai obtendo, gerando assim novas soluções parciais. Se o seu desperdício interno for superior a uma dada percentagem da folha de *stock*, essas soluções parciais são rejeitadas, não sendo mais consideradas para posteriores construções. O algoritmo terminará quando não for possível fazer nenhuma nova construção sem que se violem as restrições do problema. É então escolhida a melhor solução entre todas as geradas.

Para uma descrição mais pormenorizada do algoritmo considere-se a seguinte notação:

- β - parâmetro de rejeição, $0 \leq \beta \leq 1$;
- S_k - solução parcial gerada na iteração k ;
- $F^{(k)}$ - lista contendo todas as soluções parciais geradas na iteração k ;
- $L^{(k)}$ - lista contendo todas as soluções parciais geradas até à iteração k ;
- r_i - rectângulo do tipo i .

O algoritmo de Wang pode então ser descrito do seguinte modo:

escolha um valor para β ;

$F^{(0)} := \{r_1, r_2, \dots, r_m\}$;

$L^{(0)} := F^{(0)}$;

$k := 0$;

enquanto $F^{(k)}$ não vazio **faz**

$k := k + 1$;

$F^{(k)} := \{\}$;

 gera todas as soluções parciais S_k , fazendo construções verticais e horizontais com todos os elementos de $F^{(k-1)}$ com todos os elementos de $L^{(k-1)}$;

para cada S_k **faz**

se S_k cabe na folha de *stock*

e o número de vezes que o rectângulo i aparece em S_k não excede b_i

e o desperdício interno de S_k não é maior que βwh

então $F^{(k)} := F^{(k)} \cup S_k$;

$L^{(k)} := L^{(k-1)} \cup F^{(k)}$;

$M := k$;

escolhe o elemento de $L^{(M)}$ com menor desperdício total.

Condições suficientes de optimalidade

A dimensão do conjunto de padrões de corte de onde se escolherá o melhor - a lista $L^{(M)}$ - vai depender fundamentalmente do valor de β escolhido. Aumentando este valor, há uma tendência para obter melhores soluções, uma vez que $L^{(M)}$ aumentará, mas simultaneamente aumentará o tempo de computação e o espaço de memória necessário para armazenar $L^{(k)}$. Usando valores de β baixos melhora-se a eficiência computacional mas perde-se qualidade (medida em termos de desperdício) na solução final obtida. No extremo, quando $\beta=1$ obtem-se de certeza o padrão de

corte óptimo, uma vez que nesse caso o algoritmo é exaustivo. Contudo, o tempo e espaço de memória necessários à resolução do problema tornam-se proibitivos. Com $\beta=0$ obter-se-ia apenas as soluções cujo desperdício interno fosse nulo, ou seja, poucas ou nenhuma e muito dificilmente a solução óptima.

Poder-se-ia ser levado a pensar que se β tomar um valor diferente de um, já nada se pode concluir acerca da optimalidade do padrão de corte apresentado como solução pelo algoritmo. Contudo, é possível obter um limite superior para a distância à solução óptima da solução apresentada como a melhor pelo algoritmo, e a partir daí estabelecer condições suficientes de optimalidade para essas soluções.

Seja então:

- S - solução parcial com dimensões $w_s \times h_s$;
- D_s - desperdício interno de S ;
- D_T - desperdício total do padrão de corte que se obtém colocando S numa folha de stock de dimensões $w \times h$. Então: $D_T = wh - w_s h_s + D_s$;
- D^*_T - menor valor de desperdício total que se pode obter dadas as restrições do problema - desperdício associado ao padrão de corte óptimo ou desperdício total óptimo;
- O - conjunto dos padrões de corte que têm desperdício total D^*_T ;
- β^* - menor valor de β que ainda permite a geração de pelo menos um padrão pertencente a O .

Sendo P^* um padrão de corte pertencente a O , com desperdício interno D^*_s , é imediato que $D^*_T \geq D^*_s = \beta^* w h$.

Teorema 1

Seja P o padrão de corte seleccionado como o padrão de menor desperdício total, quando o algoritmo é aplicado com $\beta = \beta_1$. Então:

$$D_T - D^*_T \leq | D_T - \beta_1 w h |$$

Prova

Considerem-se dois casos:

- (i) Se $\beta_1 < \beta^*$ então P não pertence a O .

Então, $D_T > D^*_T \geq \beta^*wh > \beta_1wh$, ou seja

$$0 < D_T - D^*_T < D_T - \beta_1wh$$

- (ii) Se $\beta_1 \geq \beta^*$ então P pertence a O e $D_T = D^*_T$.

Isto implica que $D_T - D^*_T = 0 \leq |D_T - \beta_1wh|$

Teorema 2

Se o desperdício D_T do padrão P , obtido pelo algoritmo com $\beta = \beta_1$, satisfaz $D_T \leq \beta_1wh$, então P é um padrão de corte ótimo e $D_T = D^*_T$.

Prova

Assuma-se que $D_T \leq \beta_1wh$. Se P não pertence a O então $\beta_1 < \beta^*$ e $D^*_T > \beta_1wh$, contradizendo assim a optimalidade de D^*_T . Então P pertence a O .

Corolário

Se as dimensões de S , solução parcial que origina P , forem $w \times h$ então P é um padrão de corte ótimo e $D_T = D^*_T$.

Prova

Se S tem dimensões $w \times h$ então $D_T = D_S \leq \beta_1wh$, uma vez que P foi criado pelo algoritmo a trabalhar com $\beta = \beta_1$. Aplicando o Teorema 2 obtém-se o resultado desejado.

Teorema 3

$\beta^* \leq D_T / wh$, onde D_T é o desperdício total de um padrão de corte qualquer.

Prova

Como $D^*_T \leq D_T$, vem $\beta^*wh \leq D_T$.

Na prática, o Teorema 2 permite-nos garantir (ou não) a optimalidade de uma solução apresentada pelo algoritmo, enquanto o Teorema 3 fornece uma estimativa para o valor de β a usar para que se obtenha uma solução garantidamente ótima.

Torna-se pois claro que o valor do parâmetro β é crítico na obtenção de um compromisso entre longos tempos de execução e a capacidade de se garantir a optimalidade da solução obtida.

5.4 - O algoritmo de Wang modificado.

O algoritmo de Wang modificado [30] é o resultado de uma alteração do nível de aspiração do algoritmo de Wang, ou seja, do critério de rejeição das soluções indesejáveis. Para que se possa garantir a optimalidade de uma solução, o desperdício total D_T dessa solução deve satisfazer a condição $D_T \leq \beta wh$. Quando é vital obter soluções garantidamente óptimas, o algoritmo de Wang leva demasiado tempo e usa demasiada memória. A ideia subjacente ao algoritmo modificado é a de rejeitar, tão cedo quanto possível, aquelas soluções parciais que irão posteriormente dar origem a padrões de corte cujo desperdício total seja garantidamente maior que βwh , isto é, que não irão satisfazer as condições de optimalidade apresentadas na secção anterior. Assim, prosseguindo com a filosofia ramificação-corte do algoritmo de Wang, determina-se um limite inferior para o desperdício total final que cada solução parcial poderá vir a originar. Por outras palavras, para cada solução parcial construída (um padrão de corte em embrião), o algoritmo modificado tenta avaliar o desperdício total que, na melhor das hipóteses, o padrão respectivo irá conter. Para tal, é estimado o desperdício que a porção da folha de stock não ocupada pela solução parcial irá ainda conter. Se o valor do desperdício total - desperdício interno mais estimativa do desperdício correspondente à parte ainda não ocupada - for maior que βwh , então a solução parcial é rejeitada.

Para obter esta estimativa, resolve-se um problema não restrito, ou seja, com $b_i = \infty$, para todo o i , uma vez que neste caso o desperdício associado ao padrão de corte óptimo, nunca é superior ao desperdício do padrão de corte óptimo que se obtém quando b_i é finito. De facto, é bem sabido que num problema de minimização, o valor da solução óptima é sempre inferior ou igual ao valor da solução óptima do mesmo problema acrescido de alguma restrição.

O problema dos cortes a duas dimensões sem restrições no número de rectângulos de cada tipo a cortar, pode ser fácil e eficientemente resolvido por técnicas de programação dinâmica, usando o método proposto por Gilmore e Gomory [19]. Seja então V_M o valor máximo que se pode obter de um rectângulo de comprimento x e largura y , dado pelo algoritmo de Gilmore e Gomory. É pois possível construir uma tabela que associe um valor V_M a cada par (x,y) . Como no caso que se está a considerar, o valor de cada rectângulo é igual à sua área, o desperdício mínimo associado ao par (x,y) será dado por:

$$D_m(x,y) = xy - V_M(x,y)$$

Esta tabela é calculada uma única vez e todas as avaliações da área não ocupada por cada solução parcial reduzem-se a simples leituras da tabela.

Seja então:

- S uma solução parcial, como a representada na figura 5.4;
- βwh o desperdício máximo permitido a cada solução parcial;
- D o limite inferior para o desperdício associado à porção de folha de stock não ocupada por S ;
- D_s o desperdício interno de S .

A solução parcial satisfará então o novo nível de aspiração se:

$$D_s + D \leq \beta wh$$

onde D é calculado a partir da fórmula:

$$D = \min(D_m(w-w_s, h) + D_m(w_s, h-h_s), D_m(w, h-h_s) + D_m(w-w_s, h_s))$$

Os dois argumentos da função mínimo correspondem, respectivamente, às hipóteses de dividir a área não ocupada por S pelas linhas (a) ou (b).

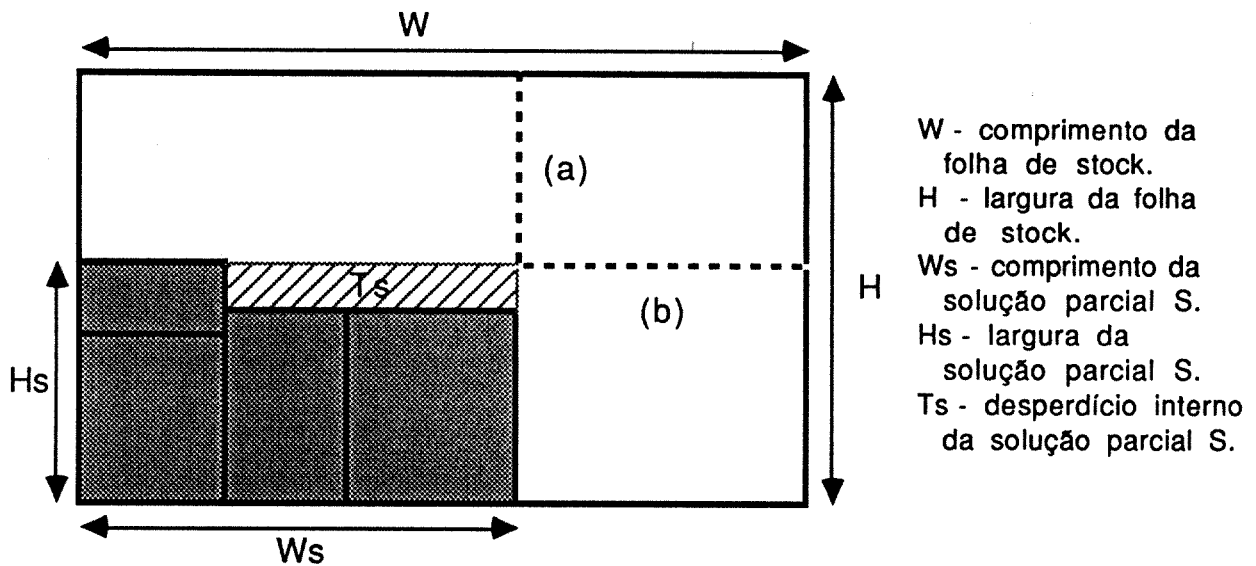


Figura 5.4 - Avaliação de uma solução parcial.

É de salientar que as condições de optimalidade apresentadas para o algoritmo de Wang se mantêm válidas para o algoritmo modificado, apesar do nível de aspiração ser alterado de $D_s \leq \beta wh$ para $D_s + D \leq \beta wh$. De facto, seja:

- D_T^* o desperdício total do padrão óptimo gerado pelo algoritmo modificado;
- β^* o menor valor de β que ainda permite a criação de um padrão óptimo.

Assuma-se que o desperdício total D_T do melhor padrão P gerado pelo algoritmo modificado, com um dado valor de β , não excede βwh . Se P não é um padrão óptimo então $\beta < \beta^*$ e as seguintes desigualdades são válidas:

$$D_T^* \geq (D_s + D)^* = \beta^* wh > \beta wh \geq D_T$$

contradizendo a optimalidade de D_T^* . Então, se $D_T \leq \beta wh$, P é um padrão se corte óptimo.

Capítulo 6

Comparação dos algoritmos: resultados computacionais.

Neste capítulo pretende-se testar o algoritmo de Wang modificado e compará-lo com o algoritmo de Wang. Pretende-se ainda comparar a eficiência do algoritmo modificado com a do algoritmo de Christofides e Whitlock. Contudo, antes da apresentação dos resultados computacionais propriamente ditos, faz-se uma descrição da estrutura da experiência efectuada, o que inclui os indicadores de desempenho, o conjunto de testes e os diferentes tipos de problemas usados. Após isso, os resultados computacionais são apresentados, comparados e algumas conclusões retiradas.

Por uma questão de simplicidade e clareza, o algoritmo de Wang será designado por AW, o algoritmo modificado por AM e o algoritmo de Christofides e Whitlock por CW.

Indicadores de desempenho

Foram escolhidos os seguintes indicadores de desempenho:

- o número de soluções parciais geradas e testadas;
- o tamanho da lista que contém os padrões finais, $L^{(M)}$;
- o desperdício gerado pelo melhor padrão;
- o número de nós expandido;
- o tempo de execução.

Os três primeiros indicadores são próprios dos AW e AM e serão usados na comparação entre esse dois algoritmos. Por sua vez, o quarto indicador é inerente ao CW.

Todas as soluções parciais, geradas pelos AW e AM, são avaliadas em termos de desperdício admissível, dimensões e satisfação das restrições ao número de rectângulos de cada tipo cortados. Como os algoritmos gastam bastante tempo a testar e rejeitar soluções parciais, pareceu interessante usar o número de soluções parciais geradas como um indicador de desempenho. Por

outro lado, este pareceu o indicador mais semelhante ao número de nós da árvore de pesquisa usado pelo CW. Assim, para além do tempo de execução, a comparação do AM com o CW será feita com base nestes dois indicadores. A qualidade de cada padrão é medida, para todos os algoritmos, pelo desperdício gerado. O tempo de execução é definido como o tempo que decorre entre o momento em que o programa inicia a resolução de um problema e o momento em que a solução é encontrada. Tempos de acesso a disco, para leitura dos dados e escrita das soluções, estão incluídos. Para a realização dos testes comparativos foi usado um SPERRY-IT, com o microprocessador INTEL 80286 e o coprocessador INTEL 80287, a trabalhar a 6 MHz. Os algoritmos foram codificados em PASCAL e compilados com o TURBO PASCAL 4, sob o sistema operativo MS-DOS, versão 3.10 .

Estrutura da experiência

Os testes computacionais efectuados podem ser divididos em duas partes: na primeira parte o AW é comparado com o AM e o desempenho de ambos os algoritmos testado, enquanto na segunda parte o AM é comparado com o CW.

A primeira parte dos testes, a comparação do AW com o AM, pode ser decomposta em três fases, de acordo com os problemas de teste usados. Na primeira fase, uma versão adaptada de três problemas apresentados por Christofides e Whitlock [9] foi usada como um conjunto de problemas estandardizado. O valor atribuído a cada rectângulo foi a sua própria área. Estes problemas possuem as seguintes características gerais:

- comprimento da folha de stock: $15 \leq w \leq 40$
- largura da folha de stock: $10 \leq h \leq 70$
- número de rectângulos diferentes: $7 \leq m \leq 20$

Para cada algoritmo, o valor de β usado foi o menor que ainda permitia a verificação das condições de optimalidade.

Para a segunda fase foi escolhido um conjunto de problemas gerado aleatoriamente por uma distribuição Beta. A área e a forma dos rectângulos foram considerados separadamente, uma vez que estes aspectos poderiam influenciar diferentemente o desempenho de ambos os algoritmos. A área A_i e a largura w_i de cada rectângulo foram gerados usando as seguintes expressões:

$$A_i = 0.25 \partial_i w h$$

$$w_i = \left(\sqrt{7A_i} - \frac{A_i}{h} \right) \partial_i + \frac{A_i}{h}$$

onde:

- A_i - área do rectângulo i ;
- w_i - largura do rectângulo i ;
- $0.25 w h$ - área máxima que um rectângulo pode ter;
- ∂_i - i -ésimo valor gerado por uma distribuição Beta com parâmetros p_1 e p_2 ;
- $0 < \partial_i \leq 1$.

O termo $\sqrt{7A_i}$ assegura que a razão comprimento/largura não excede o valor 7, enquanto o termo $\frac{A_i}{h}$ assegura que a largura do rectângulo não excede a largura da folha de *stock*. Portanto,

w_i é limitado por:

$$\frac{A_i}{h} < w_i \leq \sqrt{7A_i}$$

Finalmente, cada b_i foi amostrado de uma distribuição uniforme, num intervalo de 1 a 4. O tamanho da folha de *stock* é sempre de 70x40 e o número de rectângulos diferentes (m) é de 10 em todos os problemas.

Para testar os algoritmos AW e AM quer sob condições típicas, quer sob condições extremas, foram definidos três níveis para a área e forma dos rectângulos. Para cada nível foram usados valores bem definidos para os parâmetros da distribuição Beta [26]:

- Rectângulos sobretudo grandes - $p_1 = 8$; $p_2 = 1$
- Rectângulos com área média - $p_1 = 2$; $p_2 = 2$
- Rectângulos sobretudo pequenos - $p_1 = 1$; $p_2 = 8$
- Rectângulos sobretudo compridos - $p_1 = 11$; $p_2 = 1$
- Rectângulos com "quadratura" típica - $p_1 = 1$; $p_2 = 1$

- Rectângulos sobretudo estreitos - $p_1 = 1$; $p_2 = 11$

Foram testadas todas as combinações de área com forma. Para cada caso resolveram-se três problemas, sendo apresentados os valores médios dos resultados obtidos. Os resultados mais interessantes dizem respeito ao caso em que os rectângulos têm área média e "quadratura" típica, uma vez que esta é a situação mais comum. Para a aplicação do AW e do AM, os valores do parâmetro β usados foram os mais pequenos que ainda permitiram que cada um dos algoritmos atingisse uma solução que verificasse as condições de optimalidade.

Na terceira fase os dois algoritmos foram comparados sobre um conjunto de 10 problemas gerados aleatoriamente, com área média e "quadratura" típica.

Para a segunda parte dos testes, comparação do AM com o CW, foram utilizados estes mesmos 10 problemas, e ainda os três problemas de Christofides e Whitlock.

Resultados computacionais - comparação do AW com o AM

Os resultados obtidos na primeira fase, da primeira parte dos testes, são apresentados na tabela 6.1.

Tabela 6.1 - Resultados da primeira fase de testes

Problema	Algoritmo	β	Tempo	Nº de soluções parciais	Tamanho de $L^{(M)}$	Desperdício
1	AM	0.04	24' 03.27"	66890	623	6
	AW	0.04	32' 31.88"	75451	785	6
2	AM	0.008	20.54"	3645	80	22
	AW	0.008	2' 03.86"	19204	201	22
3	AM	0.04	9.06"	1174	73	79
	AW	0.03	1' 33.92"	10781	441	79

Neste conjunto de problemas o AM teve sempre um desempenho superior ao AW, de acordo com todos os indicadores de desempenho usados. O quociente entre o número de soluções parciais geradas varia entre 0.88 e 0.11.

Tabela 6.2 - Resultados da segunda fase de testes.

Caso	Atributos das peças	Algoritmo	Desperdício	Tempo	Nº de soluções parciais	Tamanho de $L^{(M)}$
1	Grandes e Compridas	AM	170.0	6.61"	188.7	33.3
		AW	170.0	1' 31.69"	2318.0	463.3
2	Grandes e Típicas	AM	188.0	5.60"	122.7	21.0
		AW	188.0	2.27"	295.7	85.3
3	Grandes e Estreitas	AM	201.7	5.56"	107.3	17.0
		AW	201.7	19.66"	1140.7	219.0
4	Médias e Compridas	AM	330.5	46.13"	8620.0	151.5
		AW	330.5	1h 37' 42.67"	129420.5	1307.0
5	Médias e Típicas	AM	77.7	28.9"	4149.0	121.0
		AW	77.7	3' 37.90"	16788.7	470.0
6	Médias e Estreitas	AM	118.0	32.52"	2890.0	107.5
		AW	118.0	18' 34.02"	38932.5	782.0

Os resultados da segunda fase são apresentados na tabela 6.2. Em todos os casos, excepto num, o AM demonstra ser melhor que o AW, quer em tempo de execução, quer nos outros indicadores de desempenho. A excepção ocorre no caso número 2, onde o AW apresenta um tempo de execução menor, embora nos outros indicadores seja inferior ao AM. A razão é imediata: o AM gasta um tempo fixo na inicialização do algoritmo, enquanto calcula a tabela D_m que mais tarde irá ser usada para a avaliação do desperdício de cada solução parcial. Quando os tempos de execução são muito pequenos (alguns segundos), este tempo de inicialização pode tomar uma

grande parte do tempo total. Para um problema deste tamanho (folha de *stock* com 70x40) o cálculo de D_m toma cerca de 5 segundos.

Nos casos em que os rectângulos são sobretudo pequenos, nenhum dos algoritmos atingiu a solução óptima, uma vez que o número de soluções parciais admissíveis cresce muito rapidamente, mesmo usando $\beta=0$, causando um *overflow* da memória do computador. Contudo, existem heurísticas para o problema dos cortes a duas dimensões que geralmente dão bons resultados (baixos desperdícios) nestes casos. Considerando a sua velocidade, quando se usam microcomputadores elas são uma boa alternativa para a resolução desta classe de problemas [2].

Nalguns problemas (em itálico na tabela 6.2) o AW não atingiu uma solução garantidamente óptima, enquanto o AM atingiu. De facto, o aumento do valor de β , de modo a que mais soluções parciais possam ser expandidas e as condições de optimalidade verificadas pelo AW, causa o *overflow* da memória.

Até ao momento todos os resultados computacionais apresentados foram obtidos com o menor valor de β que ainda permitia a verificação das condições de optimalidade, isto é, o valor óptimo de β . Contudo, o esforço para atingir o valor óptimo de β deve também ser considerado, sobretudo se há interesse em obter uma solução garantidamente óptima. Isto foi tomado em consideração durante a terceira fase de testes. Dez problemas típicos, gerados aleatoriamente por uma distribuição Beta com parâmetros (2,2) para a área e (1,1) para a forma, foram tomados. Cada problema foi resolvido, por ambos os algoritmos, para todos os valores de β entre 0.01 e o valor óptimo com um passo de 0.01. Os resultados são apresentados nas tabelas 6.3 a 6.6.

Tabela 6.3 - Tempos de execução (terceira fase dos testes).

		Valor de β				
Problema		0.01	0.02	0.03	0.04	0.05
1	AM	5.3"	6.2"	9.8"	13.1"	19.6"
	AW	3.0"	14.6"	54.9"	2' 18.2"	6' 5.3"
2	AM	5.3"	6.04"	8.7"	13.4"	17.9"
	AW	8.1"	35.2"	1' 05.6"	1' 46.9"	2' 13.3"
3	AM	5.3"	5.3"	8.4"	11.9"	41.0"
	AW	8.4"	1' 39.9"	10' 01.4"	41' 36.7"	2h 17' 34.5"
4	AM	5.6"	5.8"	6.8"	8.2"	10.2"
	AW	2.0"	5.8"	17.5"	45.3"	1' 23.1"
5	AM	5.9"	8.7	32.0"	3' 03.4"	
	AW	32.9"	7' 08.0"	42' 15.4"	OVERFLOW	
6	AM	5.2"	5.7"	5.8"	6.7"	
	AW	3.2"	9.5"	36.5"	1' 01.9"	
7	AM	5.8"	13.2"	41.7"		
	AW	1' 28.0"	22' 51.4"	3h 08' 21.8"		
8	AM	5.4"	5.3"	6.2"	6.2"	10.0"
	AW	2.15"	12.08"	34.4"	1' 53.4"	4' 22.4"
9	AM	5.3"	5.4"	5.4"	5.6"	5.7"
	AW	0.9"	1.8"	3.2"	6.4"	11.0"
10	AM	34.9"	12' 49.8"			
	AW	8' 02.5"	4h 25' 27.1"			

		Valor de B				
Problema		0.06	0.07	0.08	0.09	0.10
1	AM	37.6"	1' 18.2"			
	AW	15' 51.3"	38' 58.8"			
2	AM	24.3"	34.3"	50.2"	1' 10.4"	1' 30.3"
	AW	2' 53.3"	4' 01.2"	5' 24.7"	7' 12.8"	9' 25.6"
3	AM	2' 40.9"				
	AW	OVERFLOW				
4	AM	12.5"	19.3"	27.8"		
	AW	2' 34.1"	4' 49.7"	7' 49.17"		
5	AM					
	AW					
6	AM					
	AW					
7	AM					
	AW					
8	AM	16.4"				
	AW	9' 26.1"				
9	AM	6.4"	7.0"			
	AW	19.4"	30.7"			
10	AM					
	AW					

Tabela 6.4 - Número de soluções parciais (terceira fase dos testes).

		Valor de β									
Problema		0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10
1	AM	83	411	1385	2160	3736	6789	12176			
	AW	1056	3843	10376	19300	34541	58618	84795			
2	AM	102	331	871	1480	2021	2842	3931	5403	6914	8333
	AW	1478	2910	4258	6085	7501	9512	12816	15794	19292	23197
3	AM	95	95	977	1795	6326	16729				
	AW	2381	13971	39776	91329	182903	OVERFLOW				
4	AM	167	240	541	893	1405	1886	3015	4310		
	AW	557	1429	3288	5666	8215	11706	16916	27111		
5	AM	284	1204	6584	24139						
	AW	6299	34715	97353	OVERFLOW						
6	AM	79	227	266	522						
	AW	928	2262	4862	7340						
7	AM	281	2211	6412							
	AW	11771	68309	261499							
8	AM	99	99	462	462	1550	2796				
	AW	803	3476	8085	16437	27552	42050				
9	AM	87	87	145	192	269	479	659			
	AW	342	629	1044	1786	2508	3625	4598			
10	AM	6709	60586								
	AW	42415	443995								

Tabela 6.5 - Tamanho de $L^{(M)}$ (terceira fase dos testes).

		Valor de β									
Problema		0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10
1	AM	10	23	48	64	94	138	197			
	AW	52	116	233	374	599	944	1357			
2	AM	12	27	64	105	133	164	198	242	284	393
	AW	114	245	331	399	445	502	604	719	823	962
3	AM	10	10	39	57	125	246				
	AW	74	242	483	945	1663	OVERFLOW				
4	AM	17	21	34	45	62	80	113	142		
	AW	49	84	148	231	311	414	546	688		
5	AM	18	42	122	24139						
	AW	156	476	1000	OVERFLOW						
6	AM	10	20	22	38						
	AW	57	108	228	310						
7	AM	18	61	122							
	AW	197	646	1664							
8	AM	10	10	25	25	55	85				
	AW	39	107	207	360	629	964				
9	AM	10	10	15	18	24	35	46			
	AW	30	50	75	116	167	238	311			
10	AM	105	390								
	AW	369	1618								

Tabela 6.6 - Desperdícios gerados (terceira fase de testes).

Problema		Valor de β									
		0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10
1	AM	2176	1910	1190	1190	670	670	178			
	AW	488	488	292	178	178	178	178			
2	AM	2240	1292	1292	1225	1083	1983	1053	949	809	276
	AW	1136	1136	1136	523	523	435	389	343	313	276
3	AM	2155	2155	1144	648	648	147				
	AW	304	232	184	147	147	OVERFLOW				
4	AM	1400	740	740	740	740	740	565	200		
	AW	708	708	708	708	682	376	376	200		
5	AM	2032	1264	262	104						
	AW	520	122	122	OVERFLOW						
6	AM	2140	966	966	94						
	AW	160	94	94	94						
7	AM	2036	390	80							
	AW	390	80	80							
8	AM	2305	2305	1474	1474	1032	160				
	AW	1032	328	162	162	162	160				
9	AM	2170	2170	1160	1160	1160	592	192			
	AW	460	460	440	440	440	192	192			
10	AM	1080	50								
	AW	56	50								

Em primeiro lugar, é de salientar que os resultados das tabelas 6.3 a 6.6 confirmam que, para o mesmo β , o AM é mais rápido e gera menos soluções parciais que o AW. Da tabela 6.6 torna-se claro que o AM gera padrões com desperdício maior que aqueles gerados pelo AW. Isto era esperado, uma vez que o AM foi preparado para só aceitar soluções parciais que possam vir a

originar padrões de corte óptimos, sendo por isso muito mais exigente e rejeitando soluções parciais que, possivelmente, iriam gerar padrões bons mas não óptimos. Por outro lado deve ser realçado que pode acontecer que, embora atingindo a solução óptima, o AW não consiga garantir a sua optimalidade. Se β é aumentado de forma a que as condições de optimalidade sejam satisfeitas, então ocorre um *overflow* de memória (ver problemas 3 e 5 da terceira fase de testes).

Contudo, a eficiência global de ambos os algoritmos depende da rapidez com que o valor óptimo de β pode ser atingido. Mais uma vez o AM provou ser superior ao AW. Wang [34] propôs um procedimento iterativo para encontrar o valor óptimo de β , baseado no desperdício total de um padrão de corte previamente gerado. A viabilidade deste procedimento depende da qualidade, em termos de desperdício, do padrão inicial, uma vez que desperdícios totais elevados resultam em valores impraticáveis para β . Por esta razão, juntamente com o facto de não se estarem a considerar padrões iniciais externos, o procedimento iterativo proposto por Wang não foi aplicado ao AM. Em alternativa implementou-se um procedimento iterativo simples, que começa com $\beta=0.01$ e vai sucessivamente aumentando o seu valor de 0.01, até que a solução óptima seja atingida. A figura 6.1 ilustra o tempo total que o AM levaria a resolver cada um dos 10 problemas gerados para a terceira fase destes testes, usando este procedimento iterativo simples, *versus* o tempo que o AW gastaria apenas na iteração que gera a solução garantidamente óptima, isto é, supondo que de algum modo o valor óptimo de β , para o AW, era antecipadamente conhecido.

Da figura 6.1 surge claramente que o AM leva muito menos tempo no processo iterativo de encontrar o valor óptimo de β , que o AW só na iteração que garante a optimalidade da solução, exceptuando no problema 9 onde o processo iterativo levou apenas mais 10 segundos. Se se considerar que o AW necessita de mais que uma iteração para alcançar a solução óptima (o que é extremamente provável), então a diferença cresce ainda mais. Estes resultados confirmam que o AM reduz significativamente o tempo necessário à obtenção da solução óptima.

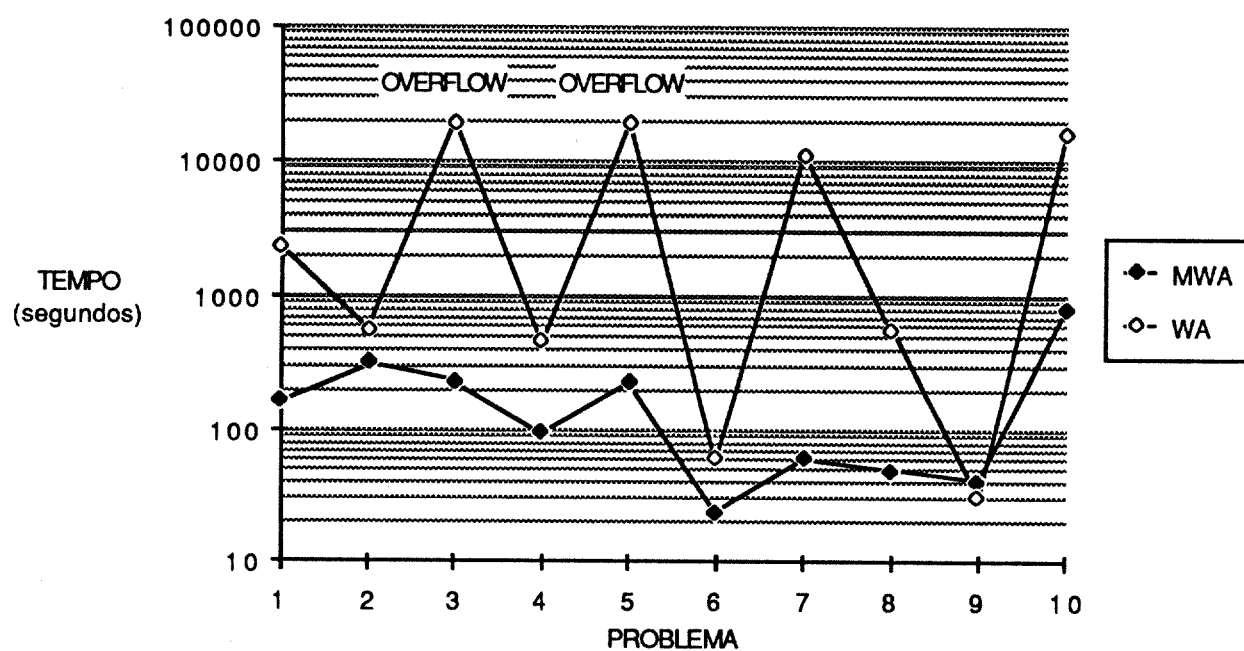


Figura 6.1 - Procedimento iterativo do AM versus iteração do AW com o β ótimo.

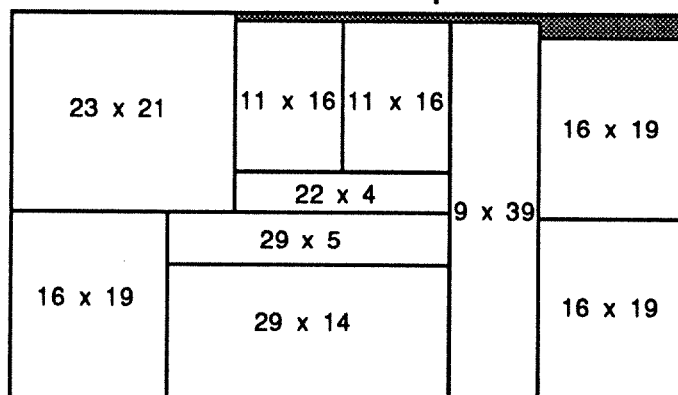
Tabela 6.7 - Dados relativos a dois dos problemas usados na segunda fase dos testes.

Problema 1				Problema 2							
i	w _i	h _i	b _i	i	w _i	h _i	b _i	i	w _i	h _i	b _i
1	29	5	1	6	23	21	3	1	22	18	2
2	9	39	4	7	29	14	4	2	0	10	1
3	55	9	1	8	16	19	3	3	13	27	3
4	31	15	1	9	9	36	2	4	23	18	2
5	11	16	2	10	22	4	2	5	29	8	4
								6	16	4	1
								7	47	9	1
								8	19	19	4
								9	13	16	2
								10	36	16	4

Na tabela 6.7 são apresentados os dados respeitantes a dois dos problemas gerados para a segunda fase dos testes (caso 5 da tabela 6.2). Os padrões obtidos (por ambos os algoritmos) encontram-se na figura 6.2.

Problema 1

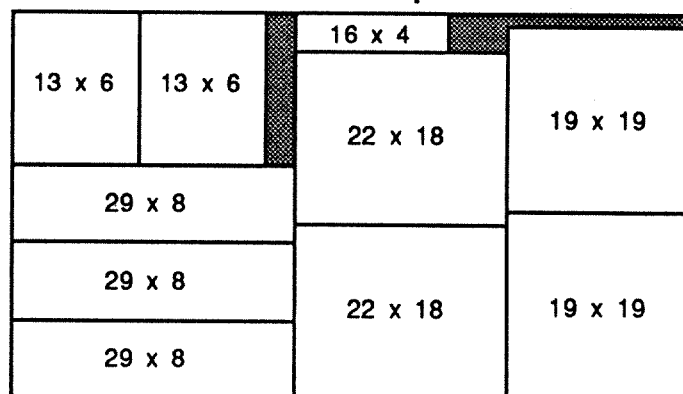
MWA e WA **Desperdício = 63**



SOLUÇÃO OPTIMA

Problema 2

MWA e WA **Desperdício = 110**



SOLUÇÃO OPTIMA

Figura 6.2 - Soluções obtidas para os dois problemas apresentados como exemplo.

Resultados computacionais - comparação do AM com o CW

Para comparar a eficiência do algoritmo modificado (AM) com o algoritmo de Christofides e Whitlock (CW), usou-se este último algoritmo para resolver dois dos conjuntos de problemas já usados na comparação do algoritmo de Wang com o algoritmo modificado.

Na primeira fase resolveram-se os três problemas apresentados por Christofides e Whitlock [9]. Para estes problemas, que surgem no artigo citado como teste computacional ao CW, os autores apresentam os respectivos resultados: valor da solução ótima, tempo de resolução (num CDC 7600) e o número de nós da árvore de pesquisa. Estes valores podem ser consultados na tabela 6.8. Ao implementarmos o CW procuramos seguir fielmente, a menos das mudanças de estruturas a que as diferentes linguagens de programação obrigam, a descrição do algoritmo que é apresentada no apêndice desse artigo.

Tabela 6.8 - Resultados computacionais apresentados por Christofides e Whitlock.

Problema	Valor da solução ótima	Tempo de resolução	Número de nós
1	244	2.47"	3794
2	2892	24.07"	18602
3	1860	66.14"	22184

Os resultados que nós obtivemos com este conjunto de problemas foram surpreendentes, mas pela negativa. Assim:

- no problema 1, o algoritmo CW não foi capaz de encontrar a solução ótima em tempo útil, isto é, interrompemos a sua resolução ao fim de 24 horas, sem que ela tivesse terminado;
- no problema 2, a solução ótima foi encontrada ao fim de 24' 07.58", gerando uma árvore com 65827 nós, o que é bastante pior que os resultados obtidos pelo AM para o mesmo problema: 20.54" e 3645 soluções parciais;

- no problema 3, a solução óptima foi encontrada ao fim de 6h 51' 51.00", através de uma árvore de pesquisa com 435033 nós, o que é novamente muito pior que os resultados obtidos pelo AM.

Acerca destes resultados são de realçar dois aspectos. Por um lado, os tempos apresentados para os problemas 2 e 3 dizem respeito unicamente à pesquisa em árvore, não sendo considerado o tempo de inicialização do algoritmo, nomeadamente o tempo de geração dos conjuntos com os pontos de corte admissíveis. Este tempo também não é completamente desprezável (num caso limite, o problema 3, este tempo rondou as 10 horas !!). O outro aspecto tem que ver com o número de nós da árvore de pesquisa. Por comparação com os resultados de Christofides, constata-se que as árvores por nós obtidas têm muito mais nós que aquelas obtidas por Christofides. Uma vez que todos os mecanismos da pesquisa em árvore propostos por Christofides foram implementados e testados cuidadosamente, até como consequência dos maus resultados obtidos, não encontramos qualquer razão para a discrepância destes números, que inclusivé são independentes da qualidade de software. Adiantamos apenas a hipótese de Christofides não descrever completamente o algoritmo, faltando algum mecanismo que aumente drasticamente a eficiência do algoritmo CW. É de realçar que, embora o algoritmo CW seja citado muitas vezes na literatura como sendo um marco importante na resolução do problema de cortes 2D uni-padrão, sobretudo pelos conceitos introduzidos, não temos conhecimento de qualquer trabalho onde ele tenha sido implementado, testado e os seus resultados confrontados com os apresentados pelos autores.

Na segunda fase de testes usou-se o algoritmo CW para resolver os 10 problemas aleatórios típicos já utilizados na comparação do AM com o AW. Os resultados apresentados na tabela 6.9 devem ser confrontados com os resultados constantes nas tabelas 6.3 a 6.6.

Tabela 6.9 - Resultados obtidos na segunda fase de testes pelo algoritmo CW.

Problema	Valor da solução ótima	Tempo de resolução	Número de nós
1	abortado		
2	2524	1' 37.77"	4651
3	abortado		
4	2600	48' 36.45"	113162
5	abortado		
6	2702	4' 07.53"	6069
7	abortado		
8	2640	1h 24' 49.13"	86569
9	2608	1h 07' 03.28"	88656
10	abortado		

Neste conjunto de problemas, abortamos a sua resolução quando ao fim de 3 horas a solução ótima ainda não tinha sido atingida. Foi o que aconteceu nos problemas 1, 3, 5, 7 e 10. Estes resultados vem apenas confirmar o que já se tinha entrevisto na primeira fase de testes: a ineficiência do algoritmo de CW. Mesmo naqueles problemas que o CW conseguiu resolver, a sua eficiência foi bastante inferior à do AM.

Procurou-se identificar o tipo de problemas em que o algoritmo CW falhava completamente. Assim, de uma análise cuidada do comportamento do algoritmo em cada um destes problemas, pôde-se concluir o que já era previsível: que o algoritmo CW vê o seu desempenho muito prejudicado quando alguma das peças a cortar é bastante mais pequena que a chapa de *stock*. E isto por duas ordens de razões. Por um lado porque os conjuntos de pontos de corte admissíveis ficam bastantes grandes, originando muitas hipóteses de corte e uma árvore de pesquisa muito grande. Por outro lado, ao retalhar a chapa de *stock* nas peças pequenas (o CW corta no sentido crescente

das coordenadas) origina problemas de transportes bastante grandes, ocasionando iterações demoradas. Estes dois factores, com a mesma causa, poderão ocorrer em simultâneo ou independentemente. Desta análise ressalta a necessidade de efectuar algumas alterações ao algoritmo, nomeadamente na sequência pela qual os cortes são executados e na criação de estratégias de ramificação que evitem ao máximo o alargamento da árvore de pesquisa.

TÉCNICAS MULTI-PADRÃO

Capítulo 7

A aplicação da Programação Linear com geração atrasada de colunas ao Problema dos Cortes.

Na Parte II desta dissertação foram analisados alguns algoritmos que resolviam o Problema dos Cortes na sua forma uni-padrão. Este tipo de algoritmos pode ser usado para resolver problemas multi-padrão através da geração e selecção sucessivas de padrões de corte, até que todos os pedidos sejam satisfeitos. Apesar de cada padrão de corte gerado ser óptimo face ao conjunto de pedidos existentes no momento, uma sucessão de padrões óptimos não garante uma solução óptima do ponto de vista global, isto é, considerando todos os pedidos em simultâneo.

A resolução do Problema dos Cortes, na forma multi-padrão, consiste então em gerar todos os padrões de corte admissíveis, em que a admissibilidade de um padrão depende das restrições que se introduzam em cada caso concreto, e determinar quantas vezes cada um desses padrões deve ser utilizado, de forma a satisfazer os pedidos existentes e a minimizar o número de chapas de *stock* gastas na satisfação desses pedidos.

Procurando formular matematicamente este problema chega-se, praticamente por transposição directa, a um modelo de Programação Linear (PL) com a seguinte estrutura:

$$\min \sum_j x_j \quad (7.1)$$

sujeito a:

$$\sum_j a_{ij} x_j = b_i, \quad i=1, \dots, m$$

$$x_j \geq 0 \text{ e inteiro}, \quad j=1, \dots, n$$

em que:

- x_j - número de vezes que o padrão de corte j é utilizado;
- b_i - número de peças do tipo i que deve ser cortado;
- a_{ij} - número de peças do tipo i que o padrão de corte j gera;
- $[a_{1j}, \dots, a_{mj}]^T$ - coluna j da matriz das restrições = padrão de corte j ;

m - número de peças diferentes a cortar;

n - número de padrões de corte.

É de salientar que o modelo apresentado não inclui qualquer consideração do tipo geométrico, pois supõe que os padrões de corte já estão gerados, isto é, que as colunas do problema já estão definidas. O modelo é portanto válido para Problemas de Cortes de qualquer dimensão.

Este modelo apresenta contudo duas dificuldades de ordem prática. A primeira delas tem a ver com a integralidade das soluções. A exigência de integralidade obriga a recorrer a técnicas de resolução que, mesmo para problemas relativamente pequenos, tornam esta formulação completamente inviável. Por esta razão, é habitual relaxar o problema, admitindo soluções não inteiras e arredondando-as no fim [16].

Contudo, para que se obtenha a solução óptima, mesmo considerando o modelo relaxado, é necessário que estejam presentes no problema todos os padrões de corte admissíveis, isto é, todos os padrões de corte que respeitam as restrições do problema. Dado o carácter combinatório da geração de padrões, o número de padrões admissíveis é sempre muito elevado, mesmo para problemas aparentemente pequenos, originando um problema de PL com um número intratável de colunas (os padrões de corte).

É neste contexto que Gilmore e Gomory, [16] e [17], sugerem a aplicação da técnica de geração atrasada de colunas à resolução dos Problemas de Cortes. Esta técnica arranca o problema de PL apenas com algumas das muitas colunas possíveis, e em cada iteração, se tal for vantajoso, cria e insere uma nova coluna no problema. Esta abordagem baseia-se na utilização do método SIMPLEX para a resolução do problema de PL e utiliza os custos marginais das variáveis não básicas da forma que seguidamente se apresenta.

Num problema de minimização (no caso presente, o modelo 7.1 sem as restrições de integralidade), em cada iteração do método Simplex entra na base a variável não básica que tiver um custo marginal mais negativo. O custo marginal de uma variável não básica é dado pela expressão:

$$\bar{c}_j = 1 - \sum_i \pi_i a_{ij}$$

em que π_i são os multiplicadores para cada uma das restrições, isto é:

$$B^T \Pi = C_B$$

em que:

B - matriz base;

Π - vector dos multiplicadores, π_i ;

C_B - coeficientes das variáveis básicas.

O problema de minimizar $\bar{c}_j$ sobre todas as colunas possíveis, quer as que já estão no problema quer aquelas que não foram ainda geradas, transforma-se no problema de maximizar:

$$\sum_i \pi_i a_{ij}$$

sob a restrição de $[a_{1j}, \dots, a_{mj}]^T$ descrever um padrão de corte admissível.

Considerando:

w_p - comprimento do padrão de corte descrito por $[a_{1j}, \dots, a_{mj}]^T$;

h_p - largura do padrão de corte descrito por $[a_{1j}, \dots, a_{mj}]^T$;

w - comprimento da chapa de stock;

h - largura da chapa de stock;

a admissibilidade do padrão $[a_{1j}, \dots, a_{mj}]^T$ passará pelas condições:

$$w_p \leq w$$

$$h_p \leq h$$

$$a_{ij} = \text{inteiro não negativo, para todo o } i.$$

A procura da variável que em cada iteração do método SIMPLEX deve entrar na base, resulta então na resolução do seguinte problema auxiliar:

$$\max \quad V = \sum_i \pi_i a_{ij}$$

$$\text{sujeito a: } w_p \leq w$$

$$h_p \leq h$$

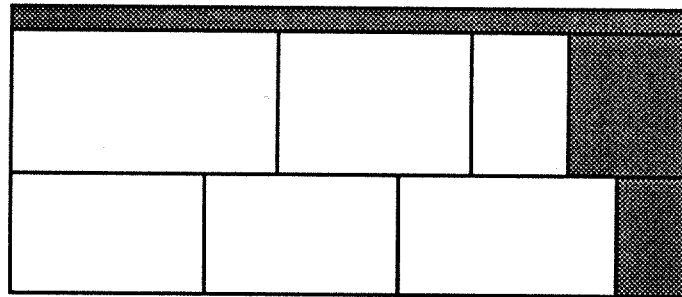
$$a_{ij} \text{ inteiros não negativos}$$

Se $V > 1$, o padrão respectivo deve ser construído e a coluna correspondente entrar na base, senão a solução corrente é ótima (todos os custos marginais são não negativos).

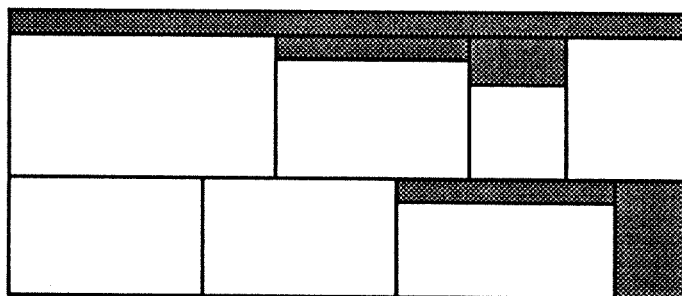
Os problemas com esta estrutura recebem o nome de problemas mochila por causa da seguinte interpretação clássica: suponha-se que se tem m objectos diferentes a transportar numa mochila; a utilidade de cada objecto é medida pelo seu valor π_i ; o conjunto de objectos a transportar não poderá exceder as dimensões da mochila, neste caso $w \times h$; o objectivo é determinar o número de objectos de cada tipo a meter na mochila, de forma a maximizar a utilidade total dos objectos transportados.

A viabilidade da abordagem da geração e introdução das colunas no problema de PL apenas se, e quando, elas melhorarem o valor da função objectivo, está sobretudo dependente da forma como se vão gerar os padrões que dão origem a custos marginais negativos, o que no caso desta abordagem proposta por Gilmore e Gomory se traduz na existência de técnicas eficientes para a resolução do problema mochila. Em [16] Gilmore e Gomory apresentam um algoritmo de Programação Dinâmica (PD) para o problema a uma dimensão. Em [17] os mesmos autores, e ainda para o caso a uma dimensão, apresentam um algoritmo de pesquisa lexicográfica que afirmam ser cinco vezes mais rápido que a técnica de PD. Em [18] os autores abordam o problema da resolução do problema mochila para o caso dos cortes a duas dimensões. Neste artigo é considerado o caso dos padrões de corte tipo guilhotina em duas ou três fases. Esta abordagem consiste basicamente em decompor a resolução do problema em duas partes: na primeira parte são geradas bandas óptimas, no sentido de maximizarem o valor das peças que se consegue encaixar numa banda com largura y , e na segunda parte as bandas são escolhidas e combinadas de forma a formarem um padrão 2D com o maior valor possível. A diferença entre a geração de padrões para duas ou três fases de corte encontra-se apenas na parte da geração das bandas óptimas. Enquanto que para padrões em duas fases se exige que todas as peças rectangulares tenham a mesma largura, para os padrões em três fases apenas se exige que a largura das peças não exceda a largura prevista para a banda (Figura 7.1 (i) e (ii)). Em ambos os casos, os dois problemas a uma dimensão que surgem (um na geração das bandas óptimas e outro na sua

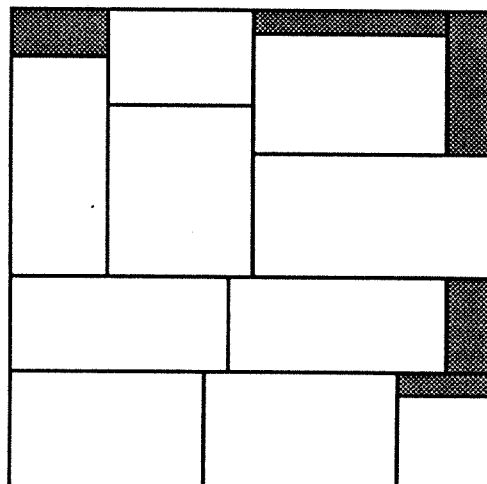
colocação) são resolvidos através do algoritmo de pesquisa lexicográfica já mencionado. No apêndice A apresenta-se uma descrição detalhada deste algoritmo.



(i) padrão em duas fases



(ii) padrão em três fases



(iii) padrão em n fases

Figura 7.1 - Padrões de corte em duas, três e n fases.

Farley, em [15], faz uma análise deste algoritmo de pesquisa lexicográfica e faz algumas adaptações, nomeadamente na geração do conjunto inicial de padrões com que a PL arranca, na consideração de soluções múltiplas por problema mochila, na valorização explícita de sub e sobreproduções de peças e na valorização das sobras, isto é, daquilo que habitualmente é encarado como desperdício.

Entretanto, em [19], Gilmore e Gomory apresentam uma rotina, baseada em PD, para a resolução do problema mochila a duas dimensões, sem mais restrições além da dos cortes serem do tipo guilhotina, ou seja, padrões em n fases (Figura 7.1 (iii)). Esta rotina, que gera padrões mais genéricos que aqueles que a dupla pesquisa lexicográfica gera, além de ser utilizada na técnica da geração atrasada de colunas, é também usada para resolver problemas de cortes uni-padrão em que não há restrições no número de peças de cada tipo cortado ([9],[30]). No Apêndice B encontra-se uma descrição detalhada desta rotina.

No capítulo 8 desta dissertação será apresentada uma abordagem algo diferente da de Gilmore e Gomory, por nós desenvolvida. Esta abordagem, é válida independentemente da técnica escolhida para a resolução dos problemas mochila que irão surgir. Na escolha da técnica de resolução dos problemas mochila deverá sempre pesar o facto da técnica de PD a duas dimensões gerar padrões mais genéricos (com menos restrições) que a dupla pesquisa lexicográfica, embora requerendo mais tempo de cálculo. Estas diferenças serão quantificadas no capítulo 9, onde serão apresentados os resultados dos testes computacionais efectuados.

Capítulo 8

Uma abordagem diferente à técnica de geração atrasada de colunas:

A geração rápida de colunas

A utilização da técnica de geração atrasada de colunas na resolução do problema dos cortes, formulado como problema de PL, se por um lado viabiliza a obtenção da solução óptima para o modelo proposto para o problema (7.1), sem implicar a geração de todos os padrões de corte admissíveis, por outro lado, ao obrigar à resolução de um problema auxiliar em cada iteração da PL, torna o custo computacional da resolução de problemas de cortes de dimensão média, extremamente elevado, quer em termos de tempo quer em termos de memória.

Os diversos autores que se têm debruçado sobre a técnica de programação linear com geração atrasada de colunas, têm procurado aumentar o seu desempenho através da procura de algoritmos mais eficientes para a resolução do problema auxiliar. Na nossa abordagem, a geração rápida de colunas, procura-se reduzir o peso da resolução do problema auxiliar diminuindo ao número de vezes que ele é resolvido.

Na programação linear com geração atrasada de colunas, como já foi discutido, para decidir em cada iteração qual a variável que deve entrar na base, seguindo o critério dos custos marginais mínimos, é necessário resolver o problema:

$$\max \sum_i \pi_i a_{ij} \quad (8.1)$$

sujeito a:

$[a_1], \dots, [a_m]$ representa um padrão de corte admissível,
para todo o j .

A admissibilidade do padrão de corte dependerá das restrições que à partida sejam impostas, ou pela própria tecnologia de corte, ou pelo facto de se pretender chegar a um modelo cuja resolução seja viável. Por exemplo, se por um lado algumas ferramentas de corte só poderão

executar padrões do tipo guilhotina, por outro lado não existe nenhuma técnica que resolva o problema 8.1 se os padrões não incluírem a restrição de serem guilhotináveis.

Posta de parte a hipótese de resolver o problema 8.1 para padrões de corte não guilhotináveis, a hipótese seguinte seria a utilização da rotina de PD, apresentada por Gilmore e Gomory [19], que gera padrões tipo guilhotina em n fases. A viabilidade da aplicação desta técnica vai contudo depender fortemente da dimensão do problema, pois envolve um número de operações $O(wh(w+h))$ e uma capacidade de armazenamento de números $O(wh)$.

Em [4] Beasley modifica esta técnica, considerando explicitamente na fórmula recursiva qual a orientação do cortes correspondente à primeira fase e utilizando o conceito de cortes normais, introduzido por Christofides e Whitlock [9], isto é, determinando o conjunto das coordenadas que são combinações lineares de comprimentos ou larguras de peças a cortar: $\{w\}$ e $\{h\}$. A determinação do padrão de corte óptimo passa a envolver $O(|w| |h| (|w| + |h|))$ operações, em que $|w|$ e $|h|$ representam o cardinal dos conjuntos $\{w\}$ e $\{h\}$, respectivamente.

Entretanto, no mesmo artigo onde apresenta esta técnica, Beasley afirma que ela se torna inviável se $\{w\}$ e/ou $\{h\}$ se tornarem grandes. Isto acontecerá se, para algum i , w_i ou h_i forem pequenos quando comparados com w ou h . Nessa situação, mesmo reduzindo o conjunto dos pontos de corte admissíveis, através da consideração de cortes normais, se w ou h forem grandes (da ordem das centenas) a técnica revela-se, do ponto de vista computacional, extremamente pesada.

Herz [23] apresentou um procedimento de pesquisa em árvore para a resolução deste mesmo problema. Segundo ele, essa técnica seria 1.5 vezes mais rápida que a rotina de PD de Gilmore e Gomory, contudo, tal como Beasley afirma [4], esse procedimento é altamente dependente da disponibilidade de um limite superior $s(x,y)$ para o valor do padrão de corte óptimo para todos os rectângulos (x,y) , $x=1,2,\dots,w$ e $y=1,2,\dots,h$. Ora se por um lado esse limite superior é facilmente obténivel quando o objectivo é maximizar a área cortada da chapa de *stock* ($s(x,y)=xy$) - tipo de problemas que Herz resolveu - por outro lado, quando o objectivo

é maximizar o valor cortado, como acontece na resolução do problema 8.1, então a tarefa de determinar o limite superior $s(x,y)$ torna-se não trivial.

Numa terceira abordagem, poder-se-ia introduzir mais uma restrição no problema auxiliar e considerar-se apenas padrões num número pré-determinado de fases, usualmente duas ou três. Computacionalmente muito menos pesada, Farley [15] pega nesta hipótese, implementa e introduz mesmo no algoritmo de pesquisa lexicográfica, apresentado por Gilmore e Gomory [17] para estes problemas, algumas capacidades adicionais. Ainda no campo da utilização de padrões em duas ou três fases, Beasley [4] implementa e modifica o algoritmo de PD de Gilmore e Gomory [16].

Contudo, e embora as técnicas que trabalham com padrões em duas ou três fases sejam viáveis para problemas razoavelmente grandes, o facto de estes padrões originarem necessariamente um desperdício igual ou superior àquele originado pelos padrões em n fases, leva a que alguns autores tentem uma resolução heurística do problema com padrões em n fases, na esperança de assim obterem uma solução melhor que aquela que um algoritmo para problemas em duas ou três fases originaria. É o caso de Beasley [4], que transforma o algoritmo recursivo para geração de padrões em n fases numa heurística, ao limitar o número de peças com que o algoritmo trabalha. Concretamente, o procedimento heurístico proposto por Beasley pode ser sintetizado da seguinte forma:

- Seja:
- $\{w\}$ - o conjunto dos pontos de corte admissíveis segundo x , construído a partir do conceito de cortes normais;
 - $|w|$ - o cardinal de $\{w\}$;
 - M - o limite superior que se pretende impôr a $|w|$, determinado a partir da eficiência e viabilidade computacional que se pretender obter;
 - N - O conjunto de peças com que o algoritmo recursivo vai trabalhar.

Então:

$N := \{1, 2, \dots, m\};$

repete

calcula $\{w\}$ a partir de N ;

se $|w| > M$ então

determina i tal que w_i é mínimo para todo $i \in N$;

$N := N - \{i\};$

até que $|w| \leq M$.

A estratégia desta heurística consiste basicamente em retirar as peças mais pequenas, que são as responsáveis pela dimensão grande do conjunto dos pontos de corte admissíveis (pontos que são combinações lineares de comprimentos/larguras das peças).

Pode-se dizer que, de uma forma geral, a perspectiva dos autores que têm abordado a questão da resolução do problema auxiliar na geração atrasada de colunas, tem sido a de encontrar técnicas mais eficientes para a resolução desse problema.

A abordagem por nós proposta, a geração rápida de colunas, assenta num princípio diferente: procura-se resolver menos vezes o problema auxiliar. A filosofia subjacente a esta abordagem não passa pela minimização do custo marginal. Assim, em cada iteração procura-se uma coluna que, entrando na base, melhore a função objectivo, e não aquela coluna que potencialmente mais melhoraria a função objectivo (aquela que possui um custo marginal mínimo). Uma consequência imediata desta abordagem é o aumento do número de iterações para atingir a solução óptima, o que implica um acréscimo de tempo na resolução do problema de PL. Contudo, se cada iteração for mais rápida, o produto *tempo por iteração* xn° de iterações poderá ser favorável. Foi sob esta permissão que a abordagem da geração rápida de colunas foi desenvolvida.

Não existindo, em cada iteração, o objectivo da minimização dos custos marginais, mas apenas o da determinação de uma variável que tenha um custo marginal negativo, convirá ainda

que, sem se procurar o custo marginal óptimo, se encontrem colunas com um custo marginal bastante negativo, isto para que o número de iterações não cresça em demasia. Para construir um padrão a que corresponda um bom custo marginal, desenvolveu-se uma heurística que gera padrões tipo guilhotina em duas, três ou n fases, conforme a opção, uma vez que a abordagem da geração rápida de colunas é perfeitamente genérica e independente do tipo de padrão em causa. Esta heurística é "glutona", pois coloca as peças por ordem decrescente de valor, isto é, começando pelas mais valiosas (π_i elevado), de forma a que

$$1 - \sum_i \pi_i a_{ij}$$

venha bastante negativo. Esta heurística será descrita pormenorizadamente no fim do presente capítulo.

Naturalmente, existirão certos momentos na resolução do problema PL em que a heurística será incapaz de gerar um padrão com custo marginal negativo. Para obter a solução óptima do problema PL é então forçoso resolver de facto o problema auxiliar da minimização dos custos marginais. Note-se, contudo, que este problema só será resolvido quando a heurística falha na tentativa de gerar um padrão de corte com um custo marginal negativo.

Resumindo, num problema de cortes, ao resolver o problema de PL, com geração rápida de colunas, para determinar em cada iteração a variável que entra na base, seguir-se-á o seguinte procedimento:

calcula $\pi_i, i=1, \dots, m$;

calcula os custos marginais das variáveis presentes no problema;

seja c_j o menor desses custos marginais e x_j a respectiva variável;

se $c_j < 0$ então x_j entra na base

senão

corre a heurística;

seja c_j o custo marginal da variável correspondente ao padrão gerado pela heurística, x_j ;

se $c_j < 0$ então x_j entra na base

senão

resolve o problema auxiliar da minimização dos custos marginais;

seja c_j o custo marginal da variável correspondente ao padrão gerado, x_j ;

se $c_j < 0$ então x_j entra na base

senão a base actual corresponde à **solução óptima**.

Heurística de geração de padrões em 3 e n fases:

Corte por Ordem Decrescente de Valores - CODV3n

Esta heurística assenta sobre o princípio de utilizar no padrão de corte as peças mais valiosas, correspondendo o valor de cada peça aos multiplicadores Simplex (π_i) de cada uma das restrições. Claramente, esta forma de proceder não garante a geração do padrão com valor máximo. Por sua vez, o funcionamento da heurística assenta na divisão sucessiva da folha de *stock*.

Ao cortar-se uma peça rectangular de uma folha rectangular maior, se se exigir que os cortes sejam do tipo guilhotina, originam-se dois novos rectângulos, a que se poderá chamar sub-folhas. Contudo, a peça poderá ser cortada de duas formas distintas: ou fazendo o primeiro

corte horizontalmente e o segundo verticalmente, ou fazendo o primeiro corte verticalmente e o segundo horizontalmente (Figura 8.1).

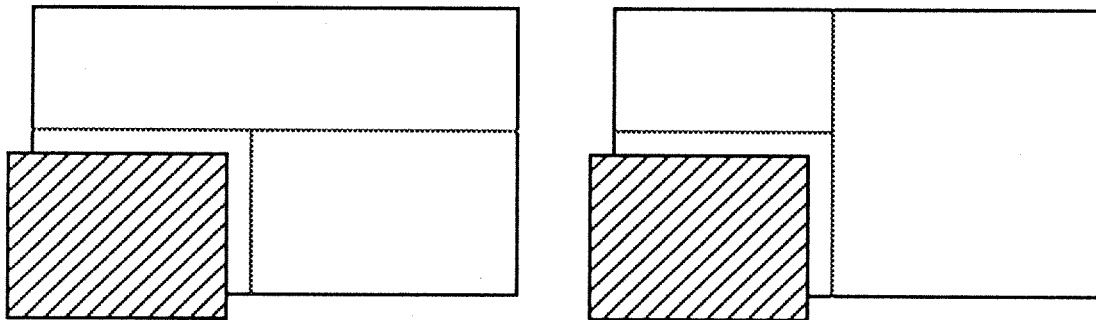


Figura 8.1 - As duas hipóteses de cortar uma peça rectangular de uma folha, usando cortes tipo guilhotina.

A heurística CODV3n, começando por ordenar as peças por ordem não crescente de valor, corta-as por essa ordem, primeiro da folha de *stock* e depois das sub-folhas que cada corte vai sucessivamente gerando, até que todas as sub-folhas sejam menores que a menor das peças. Durante este processo vai ser necessário, para cada peça cortada, determinar quais as sub-folhas geradas, o que depende do corte que é efectuado em primeiro lugar: o vertical ou o horizontal. Contudo, a decisão sobre a ordem dos cortes vai depender das restrições impostas ao padrão de corte e da geometria do corte, isto é:

- se os padrões forem em n fases então:
 - se a peça tiver comprimento/largura igual ao comprimento/largura da sub-folha de onde vai ser cortada, então apenas uma nova sub-folha será gerada - figura 8.2(i));
 - senão, serão geradas duas novas sub-folhas, efectuando-se o primeiro corte verticalmente e o segundo horizontalmente - figura 8.2(ii);
- se os padrões forem em três fases, então o primeiro corte será sempre horizontalmente. Deste modo, a sub-folha gerada pelo corte vertical só poderá conter peças com largura menor ou igual à da peça que lhe deu origem - figura 8.2(iii). A sub-folha gerada pelo corte horizontal só será considerada para cortes futuros se tiver a largura da folha de *stock*, caso contrário poderia dar origem a um padrão em mais que três fases - figura 8.3(iv), corte (6).

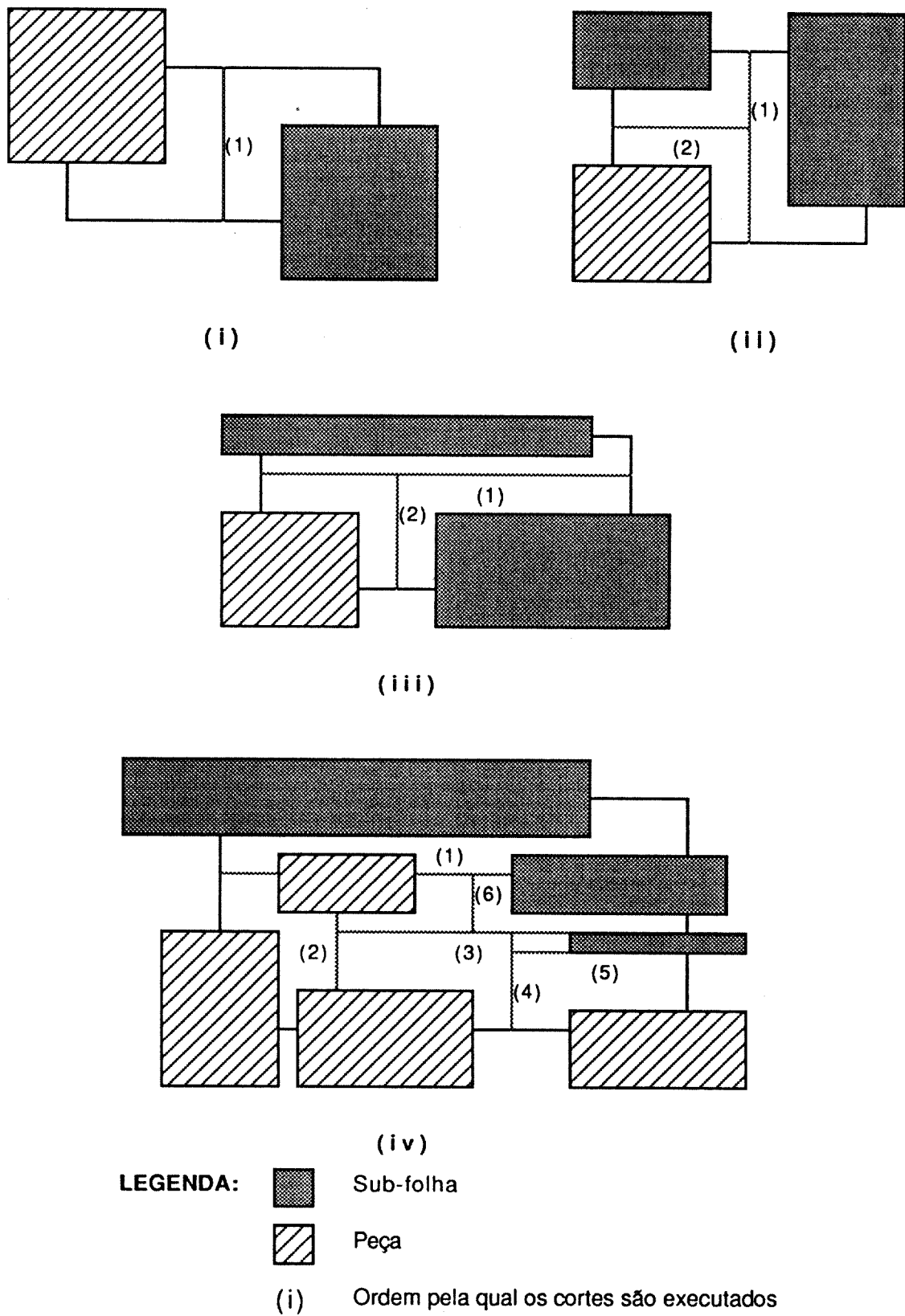


Figura 8.2 - Diferentes tipos de cortes produzidos pela heurística.

A heurística CODV3n poderá então ser representada esquematicamente da seguinte forma:

ordena as **peças** por ordem não crescente de valores;

conjunto das sub-folhas := {folha de *stock* };

repete

selecciona uma **sub-folha** e retira-a do **conjunto das sub-folhas**;

se não existir nenhuma então o **padrão** está concluído

senão

determina a **peça** mais valiosa que cabe na **sub-folha**;

se existir uma **peça** nessas condições então

corta a **peça** da **sub-folha**;

gera a(s) nova(s) **sub-folha(s)** e coloca-as no **conjunto das sub-folhas**;

até que o **conjunto das sub-folhas** = {}.

Para além das questões sobre a forma como as sub-folhas são geradas e do modo como as peças são cortadas, já abordadas nesta secção, ressalta do esquema apresentado a questão da forma como as sub-folhas são seleccionadas. À partida poder-se-ia estabelecer muitas regras, de forma a que a sub-folha seleccionada se ajustasse perfeitamente à peça a ser cortada (e que já está perfeitamente determinada, dado as peças estarem ordenadas). Contudo, os objectivos que presidiram à construção desta heurística, incluindo de facto a obtenção de um bom padrão (com um valor grande), privilegiavam a rapidez, de modo a não atrasar demasiado as iterações do problema PL. Por essa razão optou-se por não implementar nenhuma regra para a escolha das sub-folhas, com a excepção da condição da sub-folha poder conter a peça a cortar. Foi ainda por considerações de velocidade, que na geração de padrões em n fases se optou por cortar sempre em primeiro lugar verticalmente, em detrimento de uma análise das dimensões da sub-folha mãe e da peça a cortar, e de uma consequente decisão sobre a orientação do primeiro corte, de forma a obter sub-folhas filhas com melhores características, em termos de dimensões e proporções, face ao conjunto de peças a cortar.

Conforme se mostrará no próximo capítulo, a heurística CODV3n consegue de facto gerar padrões com um valor elevado, não onerando em demasia o tempo gasto nas iterações do método Simplex para a resolução do problema PL.

Capítulo 9

Estudo comparativo do desempenho da geração atrasada de colunas e da geração rápida de colunas: resultados computacionais.

Neste capítulo compara-se o desempenho da técnica clássica de Gilmore e Gomory para a resolução de problemas de cortes multi-padrão, Programação Linear com geração atrasada de colunas, com a Programação Linear com geração rápida de colunas.

A utilização de um modelo de PL para o problema dos cortes, introduz à partida uma simplificação, uma vez que um problema de cortes é inerentemente um problema inteiro. Por isso, após a resolução da PL é necessária uma outra fase, a que se chamará fase de pós-otimização, onde as soluções não inteiras são transformadas em soluções inteiras. Na primeira parte deste capítulo várias alternativas serão analisadas, sendo apresentada a abordagem escolhida para os presentes testes computacionais.

Outra questão que se coloca quando se implementa estas técnicas é a da geração do conjunto inicial de colunas/padrões para a PL. De facto, quer a geração atrasada de colunas quer a geração rápida de colunas, pressupõem a existência de uma base admissível inicial para o problema PL. Este assunto será também abordado na parte inicial deste capítulo.

Finalmente, e antes da apresentação dos resultados computacionais propriamente ditos, faz-se uma descrição da estrutura da experiência efectuada, o que inclui os indicadores de desempenho, o conjunto de testes e os diferentes tipos de problemas usados. Em seguida, e após a apresentação dos resultados computacionais, são retiradas algumas conclusões.

Por uma questão de simplicidade e clareza, a técnica de Programação Linear com geração atrasada de colunas será designada por PLGAC, e a Programação Linear com geração rápida de colunas por PLGRC.

Integralidade da solução

Obtida a solução do problema de PL, que no caso geral será não inteira, põe-se o problema de a transformar numa solução inteira, uma vez que não faz sentido dizer que um determinado padrão de corte deve ser utilizado, por exemplo, duas vezes e meia.

A forma mais imediata de obter a integralidade seria por arredondamento da solução não inteira. Contudo, o arredondamento simples das soluções, segundo a regra convencional de arredondar para cima quando a primeira casa decimal é maior ou igual a 5, e no caso contrário arredondar para baixo, faria com que a solução inteira assim obtida não verificasse as restrições do problema PL, isto é, iria dar origem a um excesso no número de peças cortadas de alguns tipos (sobreproduções), motivado pelos arredondamentos superiores, e à falta de peças de outros tipos (subproduções), motivadas pelos arredondamentos inferiores. Em processos produtivos em que sobre e subproduções sejam admitidas, esta é a forma mais prática de obter uma solução inteira.

Existem, porém, muitas situações concretas onde, embora se possam admitir sobreproduções, as subproduções são completamente inadmissíveis, uma vez que tal significaria a não satisfação de clientes ou de outros sectores dentro do mesmo processo produtivo. Nesses casos, é possível arredondar os valores não inteiros todos superiormente. Deste modo serão evitadas as subproduções, embora à custa de um aumento significativo das sobreproduções.

Em [15], Farley estuda os efeitos do arredondamento das soluções, tomando como ponto de arredondamento não só o 0.5, mas também outros valores, desde o 0.0 (todos os valores arredondados superiormente) ao 0.8 (quase todos os valores arredondados inferiormente). Nesse estudo, Farley conclui que o impacto do arredondamento depende sobretudo de dois factores:

- o número de vezes que cada peça surge num padrão de corte, e
- os pedidos existentes para cada peça.

Se o número de vezes que uma dada peça surge num padrão for elevado, pequenas alterações no número de vezes que o padrão será utilizado, levarão a grandes alterações na sobre ou subprodução dessa peça. Este facto poderia inclusivé levar à consideração de um limite superior

ao número de vezes que cada peça poderia surgir num padrão de corte. Por outro lado, se as quantidades pedidas, de uma dada peça, forem elevadas, qualquer sub ou sobreprodução será percentualmente menor e logo mais aceitável segundo todos os pontos de vista.

Quando qualquer sub ou sobreprodução é inaceitável, então estas técnicas não respondem à situação. Uma abordagem possível é o arredondamento inferior de todos os valores (truncagem) e consequente resolução de um novo problema de cortes com os pedidos não satisfeitos. Este processo seria sucessivamente repetido até que as quantidades em jogo fossem suficientemente pequenas para que o problema pudesse ser resolvido manualmente ou através de uma heurística. A desvantagem deste método reside no facto de em cada iteração serem muito provavelmente criadas soluções com cada vez maiores percentagens de desperdício, para além do tempo que tal processo iterativo tomaria. Uma alternativa a este método seria o arredondamento inferior dos valores das variáveis do problema de PL e a utilização de uma heurística, intrinsecamente inteira, para gerar os padrões que cobririam as subproduções. Um método deste tipo teria a vantagem de ser mais rápido e de fornecer bons resultados, em termos de desperdícios.

Nos testes efectuados considerou-se a situação da inadmissibilidade de sub e sobreproduções, dado ser o caso mais complexo, tendo-se optado pelo arredondamento inferior (truncagem) seguido duma heurística. A heurística escolhida foi a *First Fit Decreasing Height*. No apêndice C pode ser encontrada uma breve descrição desta heurística.

Geração dos padrões iniciais

Para que a resolução do problema de PL possa arrancar, é necessário constituir uma base admissível para o problema, devendo para tal existir um conjunto inicial de colunas/padrões. A questão que se pode colocar, é se diferentes conjuntos iniciais de padrões darão origem a diferenças significativas no tempo necessário à obtenção da solução do problema PL (com geração, atrasada ou rápida, de colunas).

Se existem m peças diferentes a cortar, então a abordagem mais simples será começar com m padrões, cada um dos quais contendo apenas uma peça, e uma peça diferente de padrão para padrão. Uma extensão desta abordagem seria incluir o maior número possível de peças de um dado tipo, em cada padrão. Tal poderia ser feito gerando riscas e depois empilhando-as sucessivamente. Contudo, as experiências mostraram que padrões constituídos por peças de um só tipo raramente são incluídos na solução óptima, dado conterem habitualmente níveis relativamente elevados de desperdício, pelo que não se justifica o acréscimo de tempo que esta abordagem provocaria na geração do conjunto dos padrões iniciais.

Farley [15] sugere uma abordagem de geração dos padrões iniciais por minimização do desperdício, isto na linha dos trabalhos realizados por Dyson e Gregory [13]. Os passos que implementariam tal abordagem seriam:

1. Defina o conjunto S como o conjunto de todas as peças pedidas.
2. Resolva um problema mochila 2D (geração de um padrão) considerando todas as peças de S e atribuindo a cada uma um valor igual à sua área.
3. Se todas as peças já foram incluídas em pelo menos um padrão então pare, senão:
4. Redefina o conjunto S como sendo todas as peças ainda não incluídas num padrão e vá para 2.

O resultado do passo 2 será um padrão contendo o menor desperdício possível, quando se considera o conjunto das peças pertencentes a S e o tipo de padrão imposto pela técnica usada para a resolução do problema mochila, nomeadamente no que diz respeito ao número de fases do padrão (nos testes efectuados foram considerados padrões em 3 e n fases). Esta abordagem produz um conjunto de padrões iniciais em que cada um contém uma combinação de várias peças, dando por isso origem a desperdícios relativamente pequenos, o que aumenta a probabilidade de virem a pertencer ao conjunto final, óptimo, de padrões. Note-se que o facto de cada peça aparecer num e num só padrão, garante a admissibilidade da solução formada por estes padrões.

A partir dos resultados de Farley [15] e da nossa própria experiência, obtida nalguns testes preliminares em que todas estas abordagens foram experimentadas, optou-se por gerar os padrões iniciais por desperdício mínimo. Embora esta abordagem seja francamente mais demorada, a diminuição no número de iterações da PL compensa o tempo gasto, sendo os tempos totais (geração dos padrões iniciais mais resolução da PL) menores do que os correspondentes à geração de padrões de peça única. A diferença de tempos dependerá sobretudo do tipo de padrão em causa, sendo mais vantajoso para os padrões em n fases, e do facto de se usar PLGAC ou PLGRC, sendo bastante mais vantajoso utilizar padrões iniciais por desperdício mínimo na PLGAC do que na PLGRC, onde por vezes chega a ser desvantajoso. De facto, a vantagem da utilização de bons padrões iniciais será tanto maior quanto mais demoradas forem; em média, as iterações da PL, o que acontecerá quando se trabalhar com padrões em n fases ou quando se usar PLGAC.

Indicadores de desempenho

Foram escolhidos os seguintes indicadores de desempenho:

- o tempo de execução (em segundos de CPU);
- o número de iterações;
- o número de padrões gerados;
- o número de chapas de *stock* gastas na solução final.

O tempo de execução, medido em segundos de CPU, refere-se ao VAX 11/750, correndo VMS, máquina onde os testes computacionais foram realizados. O programa que implementa as técnicas PLGAC e PLGRC foi escrito em PASCAL, tendo sido completamente escrito e desenvolvido por nós. Isto inclui a implementação do método SIMPLEX com geração de colunas, das técnicas de Gilmore e Gomory e das heurísticas utilizadas (CODV3n e FFDH). Sempre que possível foram utilizadas rotinas comuns para a PLGAC e a PLGRC, de modo a que os tempos de execução obtidos reflectem fielmente os diferentes desempenhos das duas técnicas. Com este mesmo objectivo, a apresentação dos tempos de execução foi dividida em duas parcelas, uma respeitante à geração dos padrões iniciais, igual para ambas as técnicas, e outra respeitante à PL propriamente dita. A fase

de pós-optimização (obtenção de uma solução inteira) foi incluída nesta segunda parcela, mas o tempo que lhe corresponde é desprezável face ao tempo de execução da PL.

Embora por si só, o número de iterações não seja um indicador relevante, uma vez que os tempos por iteração não vão ser iguais na PLGAC e na PLGRC, será interessante verificar até que ponto a geração rápida de colunas aumenta o número de iterações, assim como determinar tempos médios por iteração e assim verificar as premissas que fundamentaram o desenvolvimento da PLGRC.

O número de padrões gerados, a par com o tempo de execução, constitui um indicador de desempenho valioso e fiável. Considerando que a maior parte do tempo gasta na PL corresponde à geração de padrões, este indicador, ao medir o número de padrões gerados pela heurística CODV3n e pelas técnicas de Gilmore e Gomory, dará uma indicação sobre o desempenho da PLGAC e da PLGRC, independente do computador usado.

Finalmente, o número de chapas de *stock* gastas na solução final será o indicador que garantirá a optimalidade das soluções encontradas pela PLGRC, por comparação com os valores obtidos pela PLGAC. Note-se, contudo, que estes números poderão diferir ligeiramente, uma vez que se referem a valores posteriores à fase de pós-optimização, podendo soluções óptimas alternativas, portanto com o mesmo valor da função objectivo, ao sofrer arredondamentos e posterior pós-optimização, dar origem a alguma variação na solução inteira final.

Estrutura da experiência

Os testes computacionais efectuados foram divididos em duas fases, de acordo com os grupos de problemas utilizados.

Na primeira fase, utilizaram-se os problemas da terceira fase de testes do capítulo 6 desta dissertação, portanto problemas considerados típicos na distribuição das várias dimensões em

jogo. Foram corridos 10 problemas diferentes, todos eles com as características apresentadas na tabela 9.1.

Tabela 9.1 - Características dos problemas da primeira fase de testes.

Chapa de stock		Nº de peças diferentes (m)	Comprimento das peças (w_i)		Largura das peças (h_i)		Pedidos por peça (b_i)	
Comprimento	Largura		das peças (w_i)		das peças (h_i)		peça (b_i)	
(w)	(h)		min	max	min	max	min	max
70	40	10	6	66	3	22	10	40

Nesta fase dos testes, a PLGAC e a PLGRC foram comparadas considerando quer padrões em 3 fases, quer padrões em n fases.

Na segunda fase de testes, foram utilizados 3 conjuntos de problemas apresentados por Farley em [15] como sendo representativos de três aplicações marcadamente diferentes, por ele encontradas. Na tabela 9.2 encontram-se as características destes problemas.

Tabela 9.2 - Características dos problemas da segunda fase de testes.

Conjunto	Chapa de stock		Nº de peças		Comprimento		Largura		Pedidos por	
	Comprimento	Largura	diferentes (m)		das peças (w _i)		das peças (h _i)		peça (b _i)	
	(w)	(h)	min	max	min	max	min	max	min	max
1	200	400	20	30	20	100	20	100	100	300
2	200	400	4	10	20	100	20	100	20	100
3	200	400	20	30	50	150	50	150	3	20

Os problemas do conjunto 1 caracterizam-se por um número de pedidos por peça muito elevado e um número de peças diferentes também elevado. Por sua vez, no conjunto 2 os problemas possuem poucas peças diferentes, com um número de pedidos por peça médio. Finalmente, os problemas do conjunto 3 caracterizam-se por terem um elevado número de peças diferentes, mas com um número reduzido de pedidos por peça. Este terceiro conjunto de problemas distingue-se ainda por as peças possuírem, em média, uma área superior à dos problemas dos conjuntos 1 e 2.

Para cada conjunto foram gerados e corridos 20 problemas, tendo os valores concretos de w_i e h_i sido gerados a partir da expressão:

$$\min + (\max - \min + 1) * \sqrt{R}$$

onde R é uma variável aleatória definida sobre o intervalo $[0,1]$. Os valores de m e b_i foram amostrados de uma distribuição uniforme, sobre os limites apresentados.

Dadas as dimensões das chapas de stock , é completamente impraticável trabalhar com padrões em n fases, dado o tempo elevado necessário à resolução de cada problema mochila que viesse a surgir. Por isso, a comparação da PLGAC com a PLGRC foi feita considerando apenas padrões em 3 fases.

Resultados computacionais

Na tabela 9.3 apresentam-se os resultados obtidos nos 10 problemas da primeira fase de testes, quando se consideraram (e geraram) padrões em n fases. Por uma questão de clareza, apenas são aqui representados os valores médios dos resultados obtidos. Contudo, no Apêndice D podem ser encontrados os resultados detalhados.

Tabela 9.3 - Resultados da primeira fase de testes - padrões em n fases.

Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
			GG	Heurística	Padr. iniciais	PL
PLGAC	37.7	25.8	22.5	-	59.03"	236.18"
PLGRC	37.7	33.2	12.8	9.7	59.06"	130.74"

Deste primeiro conjunto de resultados ressalta uma nítida superioridade da PLGRC. Considerando o quociente entre os tempos médios de execução das duas técnicas, a PLGRC revelou-se cerca de 1.7 vezes mais rápida. É ainda interessante notar que, um pouco contrariamente ao que seria de esperar, a diferença entre o número de iterações, entre as duas técnicas, não é elevada, existindo mesmo alguns casos em que estes valores coincidem (ver Apêndice D). Este

facto vem de algum modo atestar a qualidade da heurística CODV3n, pois esta coincidência significa que frequentemente a heurística CODV3n gerou de facto o padrão óptimo (aquele que maximiza a função objectivo do problema auxiliar).

Este mesmo conjunto de problemas foi novamente corrido, mas agora considerando padrões em 3 fases. Os valores médios dos resultados encontram-se na tabela 9.4.

Tabela 9.4 - Resultados da primeira fase de testes - padrões em 3 fases.

Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
			GG	Heurística	Padr. iniciais	PL
PLGAC	39.2	17.0	15.2	-	0.85"	3.16"
PLGRC	39.2	23.1	8.2	8.8	0.87"	5.67"

Utilizando padrões em 3 fases os resultados invertem-se completamente: a PLGAC é francamente mais rápida que a PLGRC. O quociente entre os tempos médios de execução das duas técnicas é de 0.58. Esta diferença, face aos resultados referentes aos padrões em n fases, deve-se em primeiro lugar ao facto do algoritmo de pesquisa lexicográfica para geração de padrões em 3 fases (de Gilmore e Gomory) ser muito mais eficiente que o algoritmo de Programação Dinâmica para geração de padrões em n fases (dos mesmos autores). Por outro lado, a eficiência do algoritmo de pesquisa lexicográfica depende sobretudo do número de peças diferentes, m , em jogo, uma vez que, por cada padrão gerado, resolve $m+1$ subproblemas a uma dimensão. Ora nestes problemas da primeira fase de testes, m é relativamente pequeno. Finalmente, há que considerar que a PLGRC corre em todas as iterações a heurística CODV3n, aproveite ou não o seu resultado. Quando os tempos de execução do algoritmo de pesquisa lexicográfica e da heurística são da mesma

ordem de grandeza, cada iteração em que a heurística não seja bem sucedida, com consequente execução do algoritmo de pesquisa lexicográfica, degrada consideravelmente o desempenho da técnica. Para finalizar, deve-se considerar que se é verdade que em termos relativos as posições se invertem, em termos absolutos a PLGRC tem ainda uma grande vantagem, isto é, se neste bloco de problemas perdeu alguns segundos, no bloco anterior ganhou algumas centenas de segundos. Será ainda interessante verificar como se comportará o algoritmo de pesquisa lexicográfica quando m aumentar.

Os resultados da segunda fase de testes encontram-se na tabela 9.5. Mais uma vez apenas se apresentam os valores médios, podendo os resultados detalhados ser encontrados no Apêndice D.

Tabela 9.5 - Resultados da segunda fase de testes - padrões em 3 fases.

Conjunto de problemas	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				GG	Heurística	Padr. iniciais	PL
1	PLGAC	359.15	92.30	77.40	-	43.83"	450.00"
	PLGRC	359.25	123.10	40.35	19.00	43.67"	238.17"
2	PLGAC	33.80	18.55	16.35	-	0.55"	3.75"
	PLGRC	33.80	21.00	11.40	4.40	0.54"	6.86"
3	PLGAC	65.50	88.25	72.80	-	29.45"	356.02"
	PLGRC	65.65	107.70	26.95	26.15	29.53"	139.80"

No primeiro e terceiro conjuntos de problemas, a PLGRC mostrou-se novamente superior à PLGAC, sendo em média 1.9 e 2.5 vezes mais rápida, respectivamente. Tal como se previa, com o

aumento do número de peças diferentes (m), a eficiência do algoritmo de pesquisa lexicográfica decaiu bastante, fazendo com que a PLGRC suplantasse a PLGAC. No conjunto de problemas 2 (m pequeno) a PLGAC foi novamente a mais rápida. Contudo, tal como na primeira fase de testes, em termos globais a PLGRC mostrou-se muito superior, pois somando os tempos de execução para todos os conjuntos de problemas, obtém-se 809.77 para a PLGAC e 384.83 para a PLGRC, isto é, globalmente a PLGRC foi 2.1 vezes mais rápida.

Para além dos tempos de execução, indicador de desempenho mais importante e em função do qual a PLGRC foi desenvolvida, será curioso analisar o número de padrões gerados pelo algoritmo de Gilmore e Gomory e pela heurística CODV3n. Em primeiro lugar, o número total de padrões gerados pela PLGRC é em média inferior à PLGAC. Isto poderá mais uma vez indicar que a heurística resulta em padrões próximos da solução óptima do problema auxiliar, ou mesmo nos padrões óptimos. Poder-se-ia assim construir uma heurística para o problema global dos cortes, baseada apenas num algoritmo de PL com geração de colunas e na heurística CODV3n, para gerar esses mesmos padrões. Esta heurística global poderia ser particularmente útil quando se pretendesse trabalhar com padrões em n fases, ou para problemas com padrões em 3 fases extraordinariamente grandes.

Como conclusão, pode-se afirmar que a PLGRC é uma abordagem nitidamente mais eficiente que a PLGAC, e tanto mais eficiente quanto mais complexos forem os problemas, isto é, se se utilizarem padrões em n fases, se o tamanho das chapas de *stock* for grande, ou se o número de peças diferentes for elevado. Estes são sem dúvida os casos mais importantes. Nos problemas pequenos, dada a ordem de grandeza dos tempos envolvidos, não se justifica a sua não utilização em detrimento da PLGAC. A PLGRC é pois uma abordagem alternativa à PLGAC, cerca de 2 vezes mais rápida que esta.

Capítulo 10

Conclusões

Nesta dissertação abordamos o Problemas dos Cortes a duas dimensões, rectangular, ortogonal e em guilhotina. Em particular incidimos a nossa atenção em abordagens optimizantes, isto é, abordagens que encontram e garantem a solução óptima para os modelos considerados, quer para a forma uni-padrão do problema, quer para a forma multi-padrão.

Do estudo das técnicas uni-padrão ressaltou em primeiro lugar a comparação de técnicas de pesquisa em árvore (divisão sucessiva dos rectângulos), como o algoritmo de Christofides e Whitlock, *versus* técnicas combinatórias (combinação sucessiva dos rectângulos), o caso dos algoritmos de Wang e Wang modificado. Embora ambas as metodologias de resolução procurem limitar o seu espaço das soluções admissíveis, através da introdução de limites inferiores e/ou superiores, as técnicas de divisão dos rectângulos contêm, inerentemente à sua concepção, a desvantagem de terem de adequar os rectângulos cortados aos rectângulos pedidos. No caso do algoritmo de Christofides tal é feito através da resolução de um problema de transportes. Ora a utilização de limites superiores para o valor das soluções ainda só parcialmente obtidas, vai obrigar à resolução deste problema auxiliar em muitos dos nós da árvore de pesquisa, deteriorando assim significativamente a eficiência do algoritmo. Tal pôde ser comprovado quando se tentou usar o algoritmo de Christofides para resolver os problemas utilizados na comparação do algoritmo de Wang com o algoritmo de Wang modificado. Por outro lado, o algoritmo de Christofides é muito sensível ao critério de ramificação usado na pesquisa da árvore das soluções, pelo que seria de todo o interesse explorar novos critérios de ramificação que conduzissem a uma pesquisa mais eficiente, e consequentemente a uma árvore de pesquisa mais reduzida. Estas questões tornam-se críticas quando alguma peça a cortar é bastante mais pequena que a chapa de *stock*. Quanto à comparação entre o algoritmo por nós desenvolvido, o algoritmo de Wang modificado, e a sua versão original, o algoritmo de Wang, foi patente o aumento de eficiência. Estes algoritmos são do tipo iterativo, uma vez que necessitam de um parâmetro indicador da

percentagem de desperdício admissível, a ser usado com um nível de aspiração que as soluções parciais deverão verificar; que o valor deste parâmetro condiciona a obtenção ou não da solução óptima, o que só *a posteriori* poderá ser verificado; e que não existe nenhum meio analítico para conhecer o valor do parâmetro que permite obter a solução óptima. Assim, só resolvendo sucessivamente o mesmo problema com valores diferentes para esse parâmetro se encontrará a solução garantidamente óptima. Ora o algoritmo de Wang modificado reduz drasticamente o tempo necessário à resolução de um problema, tendo-se concluído, das experiências efectuadas, que mesmo usando um processo iterativo rudimentar, o tempo total gasto pelo algoritmo de Wang modificado para encontrar a solução óptima é, em média, bastante inferior ao tempo gasto pelo algoritmo de Wang apenas na iteração que dá origem a uma solução garantidamente óptima, isto é, admitindo-se que de algum modo se sabia o valor óptimo para o parâmetro.

Da implementação da técnica clássica de Gilmore e Gomory para a resolução de problemas multi-padrão, a programação linear com geração atrasada de colunas, concluímos o que já diversos autores afirmaram, ou seja, que a eficiência desta técnica é fortemente dependente dos algoritmos usados na resolução do problema auxiliar que surge em cada iteração do método Simplex, para que a melhor coluna (aquela com um menor custo marginal) seja gerada. Os diversos autores que abordaram esta técnica, desenvolveram os seus esforços no sentido da criação de algoritmos mais eficientes para a resolução do problema auxiliar. A nossa aposta consistiu na diminuição do número de vezes que o problema auxiliar é resolvido. Tal pode ser conseguido arranjando uma forma de gerar uma coluna boa (custo marginal negativo) embora não a melhor (o mais negativo). Para isso construímos uma nova heurística. Para que a solução óptima seja de facto encontrada, foi ainda necessário garantir que quando a heurística não fosse capaz de gerar uma coluna com um custo marginal negativo, então o problema auxiliar fosse resolvido. Isto resultou numa nova abordagem, cerca de duas vezes mais rápida que a técnica clássica: a programação linear com geração rápida de colunas. Seria bastante interessante avaliar o resultado da aplicação destes conceitos a problemas de cortes com padrões mais genéricos, isto é, com menos restrições, como por exemplo padrões não guilhotináveis ou mesmo não ortogonais. Todavia, tal só poderá ser feito após o desenvolvimento de técnicas minimamente eficientes para a

resolução dos respectivos problemas auxiliares, o que se afigura extremamente difícil. Contudo, os conceitos da geração rápida de colunas seriam aplicáveis a estas situações, sendo nossa convicção que a eficiência comparada com a técnica clássica será tanto maior quanto mais difíceis de resolver forem os problemas auxiliares.

O desenvolvimento, teste e comparação das técnicas apresentadas nesta dissertação, obrigaram a um substancial trabalho de programação. Entretanto com a programação dos algoritmos otimizantes propriamente ditos e das heurísticas por eles usadas, desenvolvidos por nós ou criados por outros autores, conjuntamente com outros nossos trabalhos [29], torna-se possível a criação de uma base de modelos/técnicas que pode evoluir para um verdadeiro sistema de apoio à decisão na área dos cortes. Este, interactivamente com o utilizador e acumulando experiência, poderá utilizar a técnica de resolução mais apropriada a cada problema concreto, tirando assim partido das virtualidades específicas de cada técnica. Esta é sem dúvida uma das extensões mais importantes do trabalho apresentado nesta dissertação.

APÊNDICES

Apêndice A

O algoritmo de pesquisa lexicográfica de Gilmore e Gomory para Problemas de Cortes a uma dimensão.

Em [17] Gilmore e Gomory apresentam um algoritmo de pesquisa lexicográfica para problema de cortes a uma dimensão, quando não há restrições no número de peças a cortar de cada tipo. Apesar de apenas gerar padrões de corte uni-dimensionais, este algoritmo é aplicado na construção de padrões de corte 2D em 2 ou 3 fases (ver capítulo 7).

Sendo então:

m	- número de comprimentos diferentes a considerar;
l_i	- comprimento i ;
s_i	- valor do comprimento i ;
L	- comprimento total da peças de <i>stock</i> ;
a_i	- número de vezes que o comprimento i é utilizado;
$a=[a_1, a_2, \dots, a_m]$	- vector representativo do padrão de corte gerar;
padrão	- melhor a em cada momento;
V	- valor de padrão ;

o algoritmo terá a seguinte estrutura:

(1) Reordenar as variáveis a_i de modo a que

$$s_1/l_1 \geq s_2/l_2 \geq \dots \geq s_m/l_m$$

(2) Fazer

$$a_1 := \text{int}(L/l_1);$$

$$a_2 := \text{int}((L - l_1 a_1)/l_2);$$

...

$$a_m := \text{int}\{ [L - (l_1 a_1 + l_2 a_2 + \dots + l_{m-1} a_{m-1})] / l_m \};$$

Fazer

$$a_1 := \sum_{l=1}^m s_l a_l;$$

$$(3) \text{ Se } L \geq \sum_{l=1}^m l_l a_l \text{ e } \sum_{l=1}^m s_l a_l > V$$

então fazer

$$V := \sum_{l=1}^m s_l a_l;$$

$$\text{padrão} := a,$$

(4) Seja e o maior i tal que $a_i \neq 0$;

(5) Fazer $a_e := a_{e-1}$;

$$\text{Se } (L - \sum_{l=1}^e l_l a_l) \cdot s_{e+1} > (V - \sum_{l=1}^e s_l a_l) \cdot l_{e+1}$$

então ir para (7) senão ir para (6);

(6) Seja e o maior i tal que $a_i \neq 0$ e $1 \leq i \leq e-1$;

Se existir tal i então ir para (5)

senão, o **padrão** actual é o óptimo (V é máximo).

(7) Fazer

$$a_{e+1} := \text{int}\{ (L - \sum_{l=1}^e l_l a_l / l_{e+1});$$

$$a_m := \text{int}\{ [(L - (\sum_{l=1}^e l_l a_l + l_{e+1} a_{e+1} + \dots + l_{m-1} a_{m-1}) / l_{e+1})] \}$$

e ir para (3).

Este algoritmo começa por colocar no padrão, o mais possível, os comprimentos com maior valor por unidade de comprimento, numa tentativa de assim obter uma solução com um valor elevado. Seguidamente, no passo (3), o algoritmo entra num ciclo em que vai retirar do padrão o comprimento com menor valor (por unidade de comprimento) - o maior i tal que $a_i \neq 0$. No passo (5), o algoritmo testa se o comprimento que sobra da peça de *stock* ao fazer-se $a_e := a_e - 1$, ao ser utilizado por comprimentos $e+1$, dá um lucro maior do que o prejuízo originado por a_e ter passado a a_{e-1} , isto é, se:

$$\frac{L - \sum_{i=1}^e l_i a_i}{V - \sum_{i=1}^e s_i a_i} > \frac{l_{e+1}}{s_{e+1}}$$

Contudo, esta desigualdade, que traduz a filosofia subjacente a este algoritmo, só será equivalente à apresentada no passo (5) do algoritmo (a que Gilmore e Gomory [17] e Farley [15] apresentam nos seus artigos) se $s_{e+1} > 0$ e $V - \sum_{i=1}^e s_i a_i > 0$, ou $s_{e+1} < 0$ e $V - \sum_{i=1}^e s_i a_i < 0$. Esta condição é

garantida se se não permitir que comprimentos com valores não positivos façam parte dos padrões de corte, isto é, se após cada atribuição a um a_i se fizer o seguinte teste:

$$\text{se } s_i \leq 0 \text{ então } a_i := 0;$$

Deste modo, a equivalência entre as desigualdades fica garantida. Por outro lado, com esta condição adicional não se está a introduzir qualquer restrição adicional aos padrões de corte, uma vez que, ao pretender-se maximizar o valor total dos comprimentos cortados pelo padrão, não faz qualquer sentido introduzir comprimentos com valores negativos.

No passo (7) do algoritmo aproveita-se o espaço libertado pelo comprimento retirado, ocupando-o com os comprimentos seguintes, gerando-se assim um novo padrão com um novo valor, supostamente melhor (tal será verificado no passo (3)). De facto não há garantia de que o novo padrão seja melhor que o anterior uma vez que a decisão de o constituir, tomada no passo (5) baseou-se no "valor acrescentado" por unidade de comprimento, o que não corresponde

necessariamente ao "valor acrescentado" final, por causa das restrições de integralidade no número de cumprimentos.

Apêndice B

O algoritmo de Programação Dinâmica de Gilmore e Gomory para Problemas de Cortes rectangulares, com padrões em n fases.

Em [19] Gilmore e Gomory apresentam um algoritmo de Programação Dinâmica (PD) para geração de padrões de cortes a duas dimensões, quando não há restrições no número de peças a cortar de cada tipo, em n fases. A justificação deste algoritmo assenta na teoria das funções mochila de duas variáveis, podendo o leitor interessado encontrar em [19] um trabalho sistemático sobre este assunto. Neste apêndice apenas se apresentará o algoritmo e procurar-se-á torná-lo verosímil na sua aplicação aos problemas de cortes 2D.

Este algoritmo gera então padrões de corte 2D, quando as peças a cortar são rectangulares, os cortes ortogonais e em guilhotina, e não há restrições no número de rectângulos a cortar de cada tipo. Não impõe restrições no número de fases de corte, pelo que se inclui no grupo dos algoritmos geradores de padrões em n fases. Uma das aplicações mais importantes deste algoritmo consiste na resolução do problema auxiliar que surge em cada iteração da resolução de problemas de cortes multi-padrão através de programação linear com geração atrasada de colunas.

Considerem-se então os seguintes conceitos e definições:

- (w, h) - dimensões da chapa de stock ;
- m - número de rectângulos diferentes a cortar;
- (w_i, h_i) - dimensões do rectângulo do tipo i , $i=1, \dots, m$;
- v_i - valor associado ao rectângulo i ;
- x_i - número de vezes que o rectângulo do tipo i é cortado;
- $F(x, y)$ - função mochila: máximo valor que se consegue obter cortando o rectângulo (x, y) em subrectângulos (w_i, h_i) , $i=1, \dots, m$;

$w(x, y)$ - menor abscissa onde é possível fazer um corte de tal modo que a soma dos valores associados aos dois subrectângulos que assim se obtêm, seja igual a $F(x, y)$;

$h(x, y)$ - idem, para as ordenadas.

O cálculo do valor da função mochila, $F(x, y)$, é feito a partir da utilização de PD:

$$F(x, y) = \max \{ F_0(x, y), F(x_1, y) + F(x_2, y), F(x, y_1) + F(x, y_2) ; \\ x \geq x_1 + x_2, 0 < x_1 \leq x_2 ; y \geq y_1 + y_2, 0 < y_1 \leq y_2 \}$$

onde:

$$F_0(x, y) = \max \{ 0, v_j ; w_j \leq x \text{ e } h_j \leq y \}$$

$F_0(x, y)$ representa o maior valor que se consegue extrair do rectângulo (x, y) sem efectuar qualquer corte, ou seja, corresponde ao valor do rectângulo mais valioso que caiba lá dentro. Se não couber nenhum, $F_0(x, y)$ valerá zero. Por sua vez, o valor da função mochila pode ser interpretado como o máximo valor que se consegue extrair do rectângulo (x, y) , optando entre não fazer corte nenhum, fazer um corte segundo x no valor da abscissa mais conveniente (que gere o par de subrectângulos mais valiosos), ou fazer um corte segundo y no valor de ordenada mais conveniente. $w(x, y)$ e $h(x, y)$ são usados para memorizar o modo como o valor $F(x, y)$ é alcançado, ou seja:

$$w(x, y) = \min \{ x_1, x ; 0 < x_1 < x - x_1 ; F(x, y) = F(x_1, y) + F(x - x_1, y) \}$$

$$h(x, y) = \min \{ y_1, y ; 0 < y_1 < y - y_1 ; F(x, y) = F(x, y_1) + F(x, y - y_1) \}$$

O algoritmo que implementa estas equações pode ser descrito da seguinte forma:

```

para  $x:=0$  a  $w-1$  faça
    para  $y:=0$  a  $h-1$  faça
         $F^*(x,y) := F_o(x,y);$ 
para  $i:=1$  a  $m$  faça
     $w^*(w_i, h_i) := w_i;$ 
     $h^*(w_i, h_i) := h_i;$ 
 $w^*(0,0) := 0;$ 
 $h^*(0,0) := 0;$ 
para  $y_2:=1$  a  $h$  faça
    para  $x_2:=1$  a  $w$  faça
        para  $x_1:=1$  a  $x_2-1$  faça
            se  $x_1+x_2 \leq w$  então
                 $V := F^*(x_1, y_2) + F^*(x_2, y_2);$ 
                se  $V \geq F^*(x_1+x_2, y_2)$  então
                     $F^*(x_1+x_2, y_2) := V;$ 
                     $w^*(x_1+x_2, y_2) := x_1;$ 
                     $h^*(x_1+x_2, y_2) := y_2;$ 
        para  $y_1:=1$  a  $y_2-1$  faça
            se  $y_1+y_2 \leq h$  então
                 $V := F^*(x_2, y_1) + F^*(x_2, y_2);$ 
                se  $V \geq F^*(x_2, y_1+y_2)$  então
                     $F^*(x_2, y_1+y_2) := V;$ 
                     $h^*(x_2, y_1+y_2) := y_1;$ 
                     $w^*(x_2, y_1+y_2) := x_2;$ 

```

Este algoritmo baseia-se num duplo ciclo em que um ponto (o ponto (x_2, y_2)) vai avançando no rectângulo (w, h) . Para cada posição que (x_2, y_2) ocupar, vão ser tentados cortes às distâncias definidas por outro ponto, o ponto (x_1, y_1) . Por outras palavras, o algoritmo, perante um rectângulo de dimensões (x_2, y_2) , encarando-o como um subrectângulo de um rectângulo de dimensões $(x_2 + x_1, y_2)$, $0 < x_1 < x_2$, primeiro e depois como um subrectângulo de um rectângulo de dimensões $(x_2, y_1 + y_2)$, $0 < y_1 < y_2$, vai decidir qual a situação mais vantajosa. No fundo, para cada rectângulo (x_2, y_2) o algoritmo decide de onde é mais vantajoso cortá-lo, se de um rectângulo $(x_2 + x_1, y_2)$ se de um rectângulo $(x_2, y_1 + y_2)$, e para que valor de x_1 ou y_1 . A figura A.1 procura esclarecer este modo de funcionamento. É de salientar o facto de x_1 ser sempre menor que x_2 e y_1 menor que y_2 . Deste modo, um rectângulo só poderá sofrer um corte até metade das suas dimensões, evitando-se assim a duplicação de padrões.

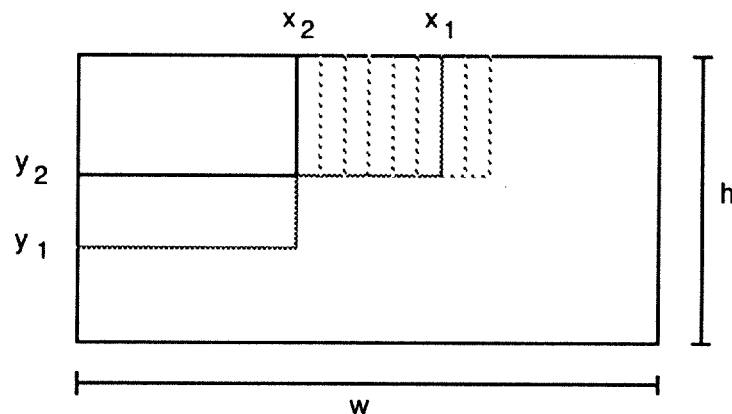


Figura A.1 - Esboço do modo de funcionamento do algoritmo 2D de Gilmore e Gomory.

No fim da execução do algoritmo, as matrizes w^* e h^* contêm a informação necessária ao levantamento do caminho seguido para chegar ao valor óptimo $F^*(x, y)$. Esse levantamento é feito através de uma pesquisa em árvore:

- se $w^*(x, y) = x$ e $h^*(x, y) = y$ então este é um nó terminal, ou seja, não se efectuou qualquer corte no rectângulo (x, y) ;

- se $w^*(x,y)=x_1 < x$ e $h^*(x,y)=y$ então deve efectuar-se um corte em x_1 e aplicar-se este processo de levantamento aos nós (x_1,y) e $(x-x_1,y)$;
- se $w^*(x,y)=y$ e $h^*(x,y)=y_1 < y$ então deve efectuar-se um corte em y_1 e aplicar-se este processo de levantamento aos nós (x,y_1) e $(x,y-y_1)$.

Gilmore e Gomory apresentam ainda uma outra versão deste algoritmo que, segundo os autores, seria mais eficiente que esta. Contudo Herz [23] mostrou que esta versão não funciona correctamente para um caso por ele apresentado.

Apêndice C

A heurística FFDH: "First Fit Decreasing Height".

A heurística "First Fit Decreasing Height" (FFDH) é uma das heurísticas clássicas usadas em problemas de empacotamentos. Embora seja utilizável em problemas de cortes (dada a dualidade entre estes dois tipos de problemas), a descrição que será feita neste apêndice usará uma nomenclatura própria dos problemas de empacotamento.

Esta heurística assume que os rectângulos a colocar estão ordenados por ordem decrescente (de facto, ordem não crescente) de altura, e coloca os rectângulos por essa ordem, formando uma sequência de níveis. Todos os rectângulos são colocados de forma ao seu fundo assentar num desses níveis. O primeiro nível será simplesmente o fundo do rectângulo grande onde os outros estão a ser colocados. Cada nível subsequente é definido por uma linha horizontal desenhada sobre o topo do primeiro (e logo o mais alto) rectângulo colocado no nível anterior.

A colocação propriamente dita funciona a partir da seguinte regra: os rectângulos são colocados encostados ao lado esquerdo do primeiro nível no qual caiba. Se nenhum dos níveis correntes acomodar este rectângulo, cria-se um novo nível onde se coloca esse rectângulo, ficando assim definida a altura desse nível.

Sejam por exemplo os seguintes 6 rectângulos (largura x altura):

- | | |
|-------------|-------------|
| - r1: 7 x 9 | - r4: 5 x 4 |
| - r2: 6 x 5 | - r5: 5 x 2 |
| - r3: 8 x 4 | - r6: 4 x 2 |

Considerando um rectângulo grande de largura 20 obter-se-ia o empacotamento representado na figura C.1.

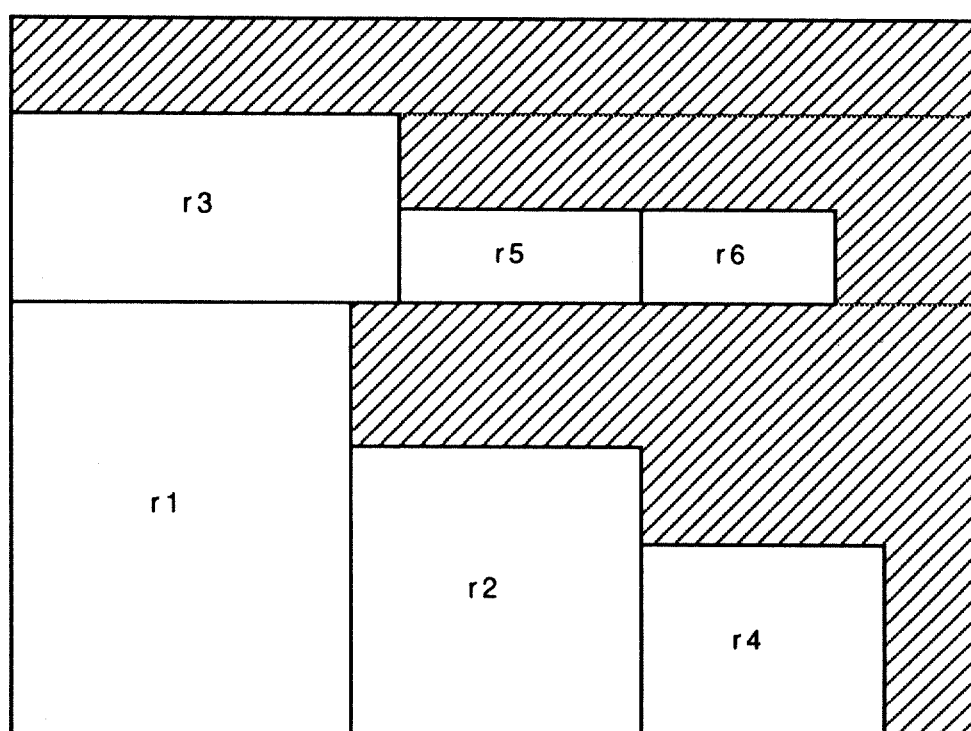


Figura C.1 - Empacotamento obtido pela heurística FFDH.

Apêndice D

Resultados detalhados dos testes computacionais efectuados no capítulo 9.

Tabela D.1 - Resultados detalhados da primeira fase de testes - padrões em n fases.

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				CG	Heurística	Padr. iniciais	PL
1	PLGAC	38	34	31	-	63.08"	310.82"
	PLGRC	38	50	22	7	63.94"	208.61"
2	PLGAC	45	11	9	-	72.72"	101.14"
	PLGRC	45	11	5	4	72.05"	57.21"
3	PLGAC	23	39	30	-	54.89"	361.86"
	PLGRC	23	47	14	16	53.97"	145.45"
4	PLGAC	41	14	13	-	78.89"	124.53"
	PLGRC	41	14	8	5	79.06"	82.93"
5	PLGAC	47	29	27	-	36.04"	262.88"
	PLGRC	48	35	5	22	36.11"	52.60"
6	PLGAC	36	21	19	-	52.77"	188.94"
	PLGRC	35	32	13	8	52.82"	131.14"
7	PLGAC	35	33	27	-	70.92"	301.28"
	PLGRC	35	38	21	5	72.10"	209.71"
8	PLGAC	26	18	17	-	53.50"	163.03"
	PLGRC	26	19	13	5	53.59"	133.09"
9	PLGAC	56	19	16	-	53.62"	173.67"
	PLGRC	56	31	10	12	52.86"	106.74"
10	PLGAC	30	40	36	-	54.02"	373.61"
	PLGRC	30	55	17	13	54.06"	179.89"

Tabela D.2 - Resultados detalhados da primeira fase de testes - padrões em 3 fases.

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
						Padr. iniciais	PL
				GG	Heurística		
1	PLGAC	39	22	21	-	0.72"	4.15"
	PLGRC	39	32	10	13	0.73"	7.13"
2	PLGAC	45	12	11	-	0.98"	2.59"
	PLGRC	45	12	8	3	0.98"	4.08"
3	PLGAC	23	15	13	-	0.98"	2.73"
	PLGRC	23	35	10	12	1.01"	7.59"
4	PLGAC	41	18	16	-	1.03"	3.52"
	PLGRC	41	17	9	5	1.00"	5.14"
5	PLGAC	52	15	13	-	0.73"	2.91"
	PLGRC	52	22	9	8	0.76"	5.69"
6	PLGAC	37	12	11	-	0.73"	1.99"
	PLGRC	37	13	5	6	0.77"	3.62"
7	PLGAC	37	17	15	-	0.76"	3.05"
	PLGRC	37	19	8	7	0.77"	5.05"
8	PLGAC	26	14	13	-	0.78"	2.68"
	PLGRC	26	12	6	5	0.87"	3.90"
9	PLGAC	58	18	16	-	0.86"	3.25"
	PLGRC	58	30	5	17	0.90"	6.26"
10	PLGAC	34	27	23	-	0.93"	4.70"
	PLGRC	34	39	12	12	0.88"	8.28"

Tabela D.3 - Resultados detalhados da segunda fase de testes, 1º conjunto de problemas - padrões em 3 fases.

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				GG	Heurística	Padr. iniciais	PL
1	PLGAC	318	84	70	-	31.94"	457.57"
	PLGRC	318	133	38	25	31.96"	215.54"
2	PLGAC	442	96	79	-	90.74"	375.12"
	PLGRC	442	124	36	20	90.78"	195.04"
3	PLGAC	451	76	68	-	193.50"	566.26"
	PLGRC	452	106	48	21	193.48"	410.43"
4	PLGAC	395	153	129	-	41.37"	823.67"
	PLGRC	395	155	51	21	40.80"	364.50"
5	PLGAC	422	84	77	-	67.21"	389.96"
	PLGRC	422	138	40	16	67.02"	240.65"
6	PLGAC	331	80	65	-	33.72"	243.44"
	PLGRC	332	131	40	19	33.40"	184.58"
7	PLGAC	322	38	36	-	17.62"	70.07"
	PLGRC	323	100	31	10	17.62"	78.47"
8	PLGAC	437	67	60	-	46.25"	514.74"
	PLGRC	437	86	32	22	46.60"	299.02"
9	PLGAC	366	118	97	-	43.43"	745.58"
	PLGRC	366	109	41	22	43.54"	280.78"
10	PLGAC	335	104	84	-	45.63"	377.52"
	PLGRC	335	123	40	18	45.24"	204.85"

Tabela D.3 (continuação)

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				GG	Heurística	Padr. iniciais	PL
11	PLGAC	322	46	42	-	14.36"	106.17"
	PLGRC	322	94	30	18	14.33"	93.73"
12	PLGAC	414	133	109	-	46.26"	1042.29"
	PLGRC	414	172	43	28	45.77"	313.74"
13	PLGAC	363	97	77	-	19.33"	390.31"
	PLGRC	363	110	43	18	19.08"	198.16"
14	PLGAC	305	90	73	-	18.15"	206.67"
	PLGRC	305	120	35	19	18.39"	118.51"
15	PLGAC	274	148	119	-	17.80"	584.07"
	PLGRC	274	158	51	19	17.66"	250.78"
16	PLGAC	429	58	51	-	58.68"	504.43"
	PLGRC	429	60	26	15	58.00"	212.60"
17	PLGAC	326	84	63	-	51.27"	385.67"
	PLGRC	326	123	46	14	50.91"	293.38"
18	PLGAC	314	93	78	-	8.31"	239.32"
	PLGRC	314	126	40	19	7.85"	129.21"
19	PLGAC	286	107	93	-	14.22"	773.29"
	PLGRC	286	153	48	21	14.12"	487.56"
20	PLGAC	331	90	78	-	16.89"	203.72"
	PLGRC	330	141	48	15	16.90"	191.96"

Tabela D.4 - Resultados detalhados da segunda fase de testes, 2º conjunto de problemas - padrões em 3 fases.

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				GG	Heurística	Padr. iniciais	PL
1	PLGAC	24	14	13	-	0.28"	2.16"
	PLGRC	24	14	10	3	0.30"	4.19"
2	PLGAC	18	16	14	-	0.29"	1.51"
	PLGRC	18	15	9	4	0.30"	3.82"
3	PLGAC	42	19	18	-	0.73"	4.66"
	PLGRC	42	18	12	5	0.70"	6.62"
4	PLGAC	22	5	4	-	0.25"	0.53"
	PLGRC	22	5	1	3	0.24"	1.31"
5	PLGAC	33	12	11	-	0.41"	1.51"
	PLGRC	33	16	8	4	0.41"	4.08"
6	PLGAC	43	15	14	-	0.55"	2.41"
	PLGRC	43	16	10	3	0.55"	4.56"
7	PLGAC	39	37	28	-	0.93"	11.80"
	PLGRC	39	40	16	7	0.92"	13.58"
8	PLGAC	34	30	27	-	0.85"	9.33"
	PLGRC	34	39	21	7	0.75"	14.42"
9	PLGAC	33	16	14	-	0.55"	2.69"
	PLGRC	33	16	11	3	0.57"	5.62"
10	PLGAC	47	25	22	-	0.49"	3.92"
	PLGRC	47	35	19	4	0.50"	9.42"

Tabela D.4 (continuação)

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				GG	Heurística	Padr. iniciais	PL
11	PLGAC	50	17	14	-	0.52"	3.08"
	PLGRC	50	23	10	4	0.49"	5.81"
12	PLGAC	46	30	28	-	0.89"	8.22"
	PLGRC	46	41	20	7	0.96"	14.64"
13	PLGAC	25	14	12	-	0.56"	2.03"
	PLGRC	25	14	8	4	0.54"	4.69"
14	PLGAC	42	7	6	-	0.62"	1.63"
	PLGRC	42	7	2	4	0.66"	2.78"
15	PLGAC	27	38	32	-	0.69"	7.90"
	PLGRC	27	38	21	7	0.62"	12.67"
16	PLGAC	28	11	10	-	0.34"	1.21"
	PLGRC	28	11	7	3	0.37"	3.46"
17	PLGAC	27	8	7	-	0.26"	0.67"
	PLGRC	27	8	4	3	0.33"	2.39"
18	PLGAC	15	17	16	-	0.35"	1.82"
	PLGRC	15	15	10	3	0.37"	4.90"
19	PLGAC	43	18	16	-	0.71"	4.17"
	PLGRC	43	21	12	5	0.72"	8.41"
20	PLGAC	38	22	21	-	0.63"	3.80"
	PLGRC	38	28	17	5	0.57"	9.90"

Tabela D.5 - Resultados detalhados da segunda fase de testes, 3º conjunto de problemas - padrões em 3 fases.

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				GG	Heurística	Padr. iniciais	PL
1	PLGAC	52	64	56	-	9.27"	100.93"
	PLGRC	53	166	25	23	9.32"	74.01"
2	PLGAC	69	90	62	-	21.24"	192.01"
	PLGRC	70	54	2	28	21.17"	21.21"
3	PLGAC	93	93	77	-	27.88"	317.88"
	PLGRC	93	141	39	34	28.10"	179.97"
4	PLGAC	58	78	62	-	12.84"	146.92"
	PLGRC	57	106	30	27	12.94"	88.66"
5	PLGAC	83	111	86	-	50.01"	653.53"
	PLGRC	83	134	44	24	50.49"	305.32"
6	PLGAC	67	147	133	-	46.06"	722.39"
	PLGRC	67	111	15	42	48.78"	107.71"
7	PLGAC	55	63	46	-	10.71"	151.74"
	PLGRC	55	85	24	27	10.53"	83.53"
8	PLGAC	85	76	64	-	35.62"	349.25"
	PLGRC	87	65	5	30	34.03"	41.12"
9	PLGAC	52	68	57	-	16.88"	161.56"
	PLGRC	52	121	31	17	16.99"	104.29"
10	PLGAC	46	80	63	-	20.44"	193.18"
	PLGRC	46	103	47	12	20.10"	135.63"

Tabela D.5 (continuação)

Problema	Técnica	Nº de chapas de stock	Nº de iterações	Nº de padrões gerados		Tempo de CPU (s)	
				GG	Heurística	Padr. iniciais	PL
11	PLGAC	72	112	92	-	47.31"	691.11"
	PLGRC	71	109	31	28	47.82"	231.55"
12	PLGAC	72	143	120	-	40.82"	984.95"
	PLGRC	72	198	30	46	40.96"	294.93"
13	PLGAC	80	62	55	-	91.62"	621.69"
	PLGRC	80	92	29	26	93.87"	331.38"
14	PLGAC	55	42	35	-	36.20"	119.06"
	PLGRC	55	50	28	9	35.59"	109.75"
15	PLGAC	72	85	71	-	25.24"	244.83"
	PLGRC	72	152	39	21	24.59"	154.19"
16	PLGAC	60	101	90	-	20.42"	428.15"
	PLGRC	60	133	32	33	20.11"	185.35"
17	PLGAC	64	75	61	-	11.80"	139.65"
	PLGRC	64	81	21	26	11.92"	60.41"
18	PLGAC	57	125	107	-	45.16"	642.36"
	PLGRC	57	122	33	27	44.05"	185.10"
19	PLGAC	50	80	65	-	8.69"	141.81"
	PLGRC	50	84	31	23	8.58"	87.17"
20	PLGAC	68	70	54	-	10.75"	117.37"
	PLGRC	69	47	3	20	10.68"	17.71"

Bibliografia

- [1] ADAMOWICZ, M. and ALBANO, A.
A Solution of the Rectangular Cutting-Stock Problem
IEEE Transactions on Systems, Man and Cybernetics 6 (1976) 4, 302-310.

 - [2] BAKER, B. S., COFFMAN, E. G. and RIVEST, R. L.
Orthogonal Packings in Two Dimensions
SIAM Journal Comput. 9 (1980) 4, 846-855.

 - [3] BAKER, B. S., BROWN, D. J. and KATSEFF, H., P.
A 5/4 Algorithm for Two-Dimensional Packing
Journal of Algorithms 2 (1981) 348-368.

 - [4] BEASLEY, J. E.
Algorithms for Unconstrained Two-Dimensional Guillotine Cutting
Journal of the Operations Research Society 36 (1985) 4, 297-306.

 - [5] BEASLEY, J. E.
An Exact Two-Dimensional Non-Guillotine Cutting Tree Search Procedure
Operations Research 33 (1985) 1, 49-64.
-

- [6] BEASLEY, J. E.
Bounds for Two-Dimensional Cutting
Journal of the Operations Research Society 36 (1985) 1, 71-74.

 - [7] BERKEY, J. O. and WANG, P. Y.
Two-Dimensional Finite Bin-Packing Algorithms
Journal of the Operational Research Society 38 (1987) 5, 423-429.

 - [8] BIRO, M. and BOROS, E.
Network Flows and Non-Guillotine Cutting Patterns
European Journal of Operational Research 16 (1984) 215-221.

 - [9] CHRISTOFIDES, N. and WHITLOCK, C.
An Algorithm for Two-Dimensional Cutting Problems
Operations Research 25 (1977) 30-44.

 - [10] COFFMAN, E. G., GAREY, M. R., JOHNSON, D. S. and TARJAN, R. E.
Performance Bounds for Level-Oriented Two-Dimensional Packing Algorithms
SIAM Journal Comput. 9 (1980) 4, 808-826.
-

- [11] DYCKHOFF, H., KRUSE, H., ABEL, D. and GAL, T.
Classification of Real World Trim Loss Problems
in: G. Fandel et al (eds), "Essays on Production Theory and Planning", Springer-Verlag,
1988, 191-208.
- [12] DYCKHOFF, H.
A Typology of Cutting and Packing Problems
EURO IX / TIMS XXVIII Paris, 1988.
- [13] DYSON, R. G. and GREGORY, A. S.
The Cutting Stock Problem in the Flat Glass Industry
Operational Research Quarterly 25 (1974) 1, 41-53.
- [14] DOWSLAND, W. B.
Two and Three Dimensional Packing Problems and Solution Methods
New Zealand Operational Research 13 (1985) 1, 1-18.
- [15] FARLEY, A. A.
Practical Adaptations os the Gilmore-Gomory Approach to Cutting Stock Problems
OR Spektrum 10 (1988) 113-123.
-

- [16] GILMORE, P. C. and GOMORY, R. E.
A Linear Programming Approach to the Cutting Stock Problem
Operations Research 9 (1961) 849-859.
- [17] GILMORE, P. C. and GOMORY, R. E.
A Linear Programming Approach to the Cutting Stock Problem - Part II
Operations Research 11 (1963) 863-888.
- [18] GILMORE, P. C. and GOMORY, R. E.
Multistage Cutting Stock Problems of Two and More Dimensions
Operations Research 13 (1965) 94-120.
- [19] GILMORE, P. C. and GOMORY, R. E.
The Theory and Computation of Knapsack Functions
Operations Research 15 (1967) 1045-1075.
- [20] GOLDEN, B. L.
Approaches to the Cutting Stock Problem
AIIE Transactions 8 (1976) 2, 265-274.
- [21] HAHN, S. G.
On the Optimal Cutting of Defective Sheets
Operations Research 16 (1968) 1100-1114.
-

- [22] HARRISON, P.

The Two-Dimensional Cutting Problem on the Furniture Manufacturer

Operational Research '84, National Contribution of the Unites Kingdom, 1984.

- [23] HERZ, J. C.

Recursive Computational Procedure for Two-Dimensional Stock Cutting

IBM Journal of Research and Development 16 (1972) 462-469.

- [24] HINXMAN, A. I.

The Trim-Loss and Assortment Problems: a Survey

European Journal of Operational Research 5 (1980) 8-18.

- [25] ISRANI, S. S. and SANDERS, J. L.

Two-Dimensional Cutting Stock Problem Research:

A Review and a New Rectangular Layout Algorithm

Journal of Manufacturing Systems 1 (1982) 2, 169-182.

- [26] ISRANI, S. S. and SANDERS, J. L.

Performance Testing of Rectangular Parts-Nesting Heuristics

International Journal of Production Research 23 (1985) 3, 437-456.

- [27] MADSEN, O. B. G.

Glass Cutting in a Small Firm

IMSOR, The Technical University of Denmark, DK-2800, Lyngby, Denmark (1979).

- [28] MADSEN, O. B. G.

A Cutting Sequencing Algorithm

IMSOR, The Technical University of Denmark, DK-2800, Lyngby, Denmark (1980).

- [29] OLIVEIRA, J. F. and FERREIRA, J. S.

Solving Two-Dimensional Cutting Problems and Comparing Different Approaches

EURO IX / TIMS XXVIII Paris, 1988.

- [30] OLIVEIRA, J. F. and FERREIRA, J. S.

An Improved Version of Wang's Algorithm for Two-Dimensional Cutting Problems

A aparecer no 1º trimestre de 1990 in:

European Journal of Operational Research - Special Issue on "Cutting and Packing Problems".

- [31] TOCZYLOWSKI, E.

On Aggregation in a Two-Dimensional Cutting Stock Scheduling Problem

Large Scale Systems 10 (1986) 165-174.

- [32] TOKUYAMA, H. and UENO, N.

The Cutting Stock Problems in the Iron and Steel Industries

Operational Research '81, National Contribution of Japan, 1981.

- [33] VASKO, F. J.

A Computational Improvement to Wang's Two-Dimensional Cutting Stock Algorithm

Computers and Industrial Engineering 16 (1989) 1, 109-115.

- [34] WANG, P. Y.

Two Algorithms for Constrained Two-Dimensional Cutting Stock Problems

Operations Research 31 (1983) 573-586.
