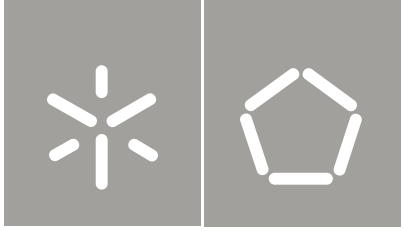

Universidade do Minho
Escola de Engenharia

Carina Maria Oliveira Pimentel

Algoritmos de partição e geração de colunas para dimensionamento de lotes de produção

Dissertação apresentada à Universidade do Minho para a obtenção do grau de Mestre em Engenharia Industrial

Trabalho efectuado sob a orientação de

Doutor José Manuel Vasconcelos Valério de Carvalho
Departamento de Produção e Sistemas da Escola de Engenharia da Universidade do Minho

Doutor Filipe Pereira Pinto da Cunha e Alvelos
Departamento de Produção e Sistemas da Escola de Engenharia da Universidade do Minho

Maio de 2005

Esta dissertação foi suportada parcialmente pela Fundação para a Ciência e a Tecnologia (Projecto POSI / 1999 / SRI / 35568) e pelo Centro de Investigação Algoritmi da Universidade do Minho.

Resumo

Nesta tese, apresentam-se duas aplicações do método de partição e geração de colunas ao problema de lotes de produção multi-artigo capacitado com tempos de preparação.

O problema de lotes de produção multi-artigo capacitado com tempos de preparação pode ser descrito como um modelo no qual se procura determinar um plano de produção para vários artigos ao longo de um determinado horizonte temporal, que minimize os custos de produção, de armazenagem e de preparação dos artigos e respeite restrições de procura e de capacidade.

A abordagem seguida é a de reformular a formulação de programação inteira mista do problema, utilizando o princípio da decomposição de Dantzig-Wolfe, e combinar o método de geração de colunas com o método de partição e avaliação (método de partição e geração de colunas) para obter soluções óptimas. Na definição da regra de partição utiliza-se uma abordagem baseada nas variáveis binárias do modelo original, que garante a preservação da estrutura do subproblema.

A partir da formulação de programação inteira mista do problema, definem-se dois modelos de decomposição: a decomposição por artigo e a decomposição por período e resolvem-se os problemas reformulados através do método de geração de colunas. Para determinar a solução óptima inteira dos problemas reformulados aplica-se o método de partição e geração de colunas.

São apresentados resultados de testes computacionais para um conjunto de instâncias com diferentes características, que permitem estabelecer comparações entre os dois modelos de decomposição. Esses resultados computacionais são ainda comparados com a formulação de programação inteira mista que se resolve através de um "solver" de problemas de programação inteira mista, o CPLEX.

Abstract

In this thesis, we present two branch-and-price algorithms to solve the multi-item capacitated lot-sizing problem with setup times.

The multi-item capacitated lot-sizing problem with setup times can be seen as a model in which we want to find a production plan for several items over a number of time periods, that minimize the labour costs, inventory costs and setup costs and satisfies all demand requirements without exceeding capacity limits.

The approach taken consists in reformulating the mixed integer problem, using the Dantzig-Wolfe decomposition principle, and then combining the column generation method with the branch-and-bound method (branch-and-price method) in order to obtain optimal solutions. In the definition of the branch rule we use an approach based on the binary variables of the original model that assure the conservation of the structure of the subproblem.

Based on the mixed integer formulation of the problem, we develop two decomposition models: the item decomposition and the period decomposition and solve the reformulated problems with the column generation method. In order to obtain the integer optimal solution of the reformulated problems we use the branch-and-price method.

We present results of the computational tests for a set of instances with different characteristics, to establish comparisons between the two decomposition models. These results are then compared with the mixed integer formulation that we solve by CPLEX, a mixed integer problems solver.

Agradecimentos

Gostaria de manifestar o meu mais sincero agradecimento a todas as pessoas e instituições que contribuíram directa ou indirectamente para que a realização deste trabalho fosse possível.

Aos meus orientadores, Professor Valério de Carvalho e Professor Filipe Alvelos, ficarei eternamente grata pela oportunidade que me deram de experimentar a investigação científica, por me terem encorajado a realizar o Mestrado em Engenharia Industrial e pelo seu apoio desde que iniciei este trabalho.

Tenho ainda a agradecer-lhes a orientação científica, a partilha de conhecimento e experiência, a confiança e a dedicação ao longo deste tempo. Por último, quero agradecer-lhes todas as sugestões feitas durante o desenvolvimento do trabalho e durante a escrita da dissertação, assim como a sua revisão final.

À Universidade do Minho, em particular ao Departamento de Produção e Sistemas e à Comissão Directiva do Mestrado em Engenharia Industrial, agradeço a oportunidade que me concederam em participar no Mestrado bem como a disponibilidade das instalações.

Ao Acácio tenho a agradecer as essenciais e descontraídas pausas e o seu bom humor, que sempre ajudaram a ganhar ânimo para continuar a trabalhar e todo o apoio técnico.

À Celeste, à Cândida e ao Pedro, pela companhia e amizade.

Ao Miguel, pelo seu apoio incondicional e pela compreensão pelos momentos em que deixei de estar com ele para conseguir realizar este projecto.

Aos meus pais e irmãos, pelo encorajamento, apoio e compreensão.

Índice

1 Introdução Geral

Nesta dissertação aborda-se o problema de lotes de produção multi-artigo capacitado com tempos de preparação (LPMACC) e apresentam-se dois métodos de resolução para esse problema.

O problema LPMACC, para além de estar relacionado com um tema de elevada relevância, é um problema que se reveste de grande complexidade, não só pelo número de variáveis envolvidas no modelo, mas também pelo tipo e número de restrições envolvidas.

Este problema corresponde ao problema resolvido ao nível do plano mestre de produção e tem sido extensivamente estudado, pelo facto de ser um problema importante do planeamento da produção. Em termos gerais, o problema LPMACC consiste na determinação de um plano de produção para vários artigos, habitualmente produtos acabados, num número finito de períodos de planeamento por forma a satisfazer a procura dos clientes, tendo em conta a escassez dos recursos, ao menor custo possível.

Existe uma enorme variedade de problemas de lotes de produção. O problema que se aborda neste trabalho pertence à classe de problemas de lotes de produção de nível único, multi-artigo, com procura determinística e dinâmica e capacitado. Muitos dos estudos sobre este problema não consideram no modelo o tempo gasto na preparação dos recursos antes que se possa iniciar a produção. Como o tempo de preparação consome parte da capacidade de produção disponível, nesta versão do problema não existe a garantia de que a solução para o problema seja admissível. Por outro lado, com a crescente preocupação em adaptar os sistemas de produção à filosofia do "Just in Time", torna-se essencial a contabilização do tempo gasto na preparação. Nos sistemas de "Just in Time" os tempos de preparação são reduzidos o mais possível para que se possam produzir lotes mais pequenos e consequentemente reduzir o nível dos em curso de fabrico ("work in progress"). Deste modo, garante-se o aumento da produtividade do sistema de produção e a redução do ciclo de fabrico dos artigos ("lead time"). No sentido de modelar correctamente o problema aqui abordado considera-se na sua formulação

o tempo de preparação.

Para resolver o problema LPMACC, a partir de uma formulação de programação inteira mista definem-se dois modelos de decomposição aplicando a decomposição de Dantzig-Wolfe (Dantzig and Wolfe 1960) e resolvem-se os problemas reformulados através do método de geração de colunas. Ao resolver os problemas reformulados pelo método de geração de colunas, não existe a garantia de obtenção de uma solução óptima inteira. Torna-se então necessário aplicar o método de partição e geração de colunas ("branch-and-price") que resulta da combinação do método de geração de colunas com o método de partição e avaliação. O método de partição e geração de colunas tem sido aplicado com sucesso a diversos problemas de optimização. A título de exemplo refere-se a sua aplicação a problemas de escalonamento de rotas de veículos (Desrosiers, Dumas et al. 1995) e ao problema de corte (Carvalho 2002), entre outros.

Contudo, a utilização deste método é relativamente recente. Em alguns artigos de revisão sobre o método de partição e geração de colunas ((Barnhart, Johnson et al. 1998), (Lübbecke and Desrosiers 2002), (Wilhelm 2001)) refere-se que existem alguns aspectos do método que devem ser mais explorados, abrindo-se horizontes em termos de futuro neste campo de investigação.

Nos parágrafos seguintes, apresentam-se as principais contribuições deste trabalho de acordo com o nosso ponto de vista.

A apresentação de dois modelos de decomposição para o problema LPMACC, baseados na decomposição de Dantzig-Wolfe (Dantzig and Wolfe 1960), que permitem obter formulações mais fortes e melhores limites inferiores para o problema.

O primeiro modelo de decomposição, que se denomina por decomposição por artigo, baseia-se na decomposição das restrições de capacidade e foi já estudado por alguns autores. O que é inovador no nosso trabalho é o desenvolvimento de novas regras de partição no método de partição e geração de colunas.

O segundo modelo de decomposição, que se denomina por decomposição por período, baseia-se na decomposição das restrições de equilíbrio de stocks. No nosso conhecimento é a primeira vez que um algoritmo de partição e geração de colunas é desenvolvido com base nesta decomposição.

A apresentação de uma metodologia geral para a resolução de problemas de programação inteira mista (para os quais é conhecida uma formulação compacta), baseada no método de partição e geração de colunas.

A utilização do método de partição e geração de colunas para determinar soluções exactas para o problema de lotes de produção multi-artigo capacitado com tempos de preparação, constituído por variáveis contínuas e binárias e a definição de uma regra de partição baseada nas variáveis originais, que garante a preservação da estrutura do subproblema.

A apresentação de alguns testes computacionais relativos à resolução do problema LPMACC utilizando os dois modelos de decomposição, que permitem estabelecer o paralelo entre duas decomposições baseadas no mesmo problema e formulação e a comparação com um "package" comercial de programação inteira mista.

O objectivo genérico deste trabalho consiste na aplicação de técnicas de partição e geração de colunas a um problema de lotes de produção com vista à obtenção de um modelo mais forte e à obtenção de melhores tempos de resolução desse problema. Os objectivos específicos a considerar serão o estudo de um modelo de programação inteira mista para o problema de lotes de produção multi-artigo capacitado com tempos de preparação, a aplicação de duas decomposições de Dantzig-Wolfe a essa formulação, a resolução dos problemas reformulados usando o método de geração de colunas e a aplicação do método de partição e geração de colunas para a obtenção da solução óptima inteira do problema. Para a realização de parte dos objectivos referidos serão implementadas duas classes de C++ onde se definirão os dois modelos de decomposição que resultam da aplicação da decomposição de Dantzig-Wolfe e utilizar-se-á o software ADDing ("Automatic Dantzig-Wolfe Decomposition for Integer Column Generation") (Alvelos 2005), que implementa um algoritmo de partição e geração de colunas.

No capítulo 2 faz-se um enquadramento geral do sistema de planeamento e controlo da produção, revêem-se e classificam-se os modelos de lotes de produção de nível único, com procura determinística e dinâmica e capacitados, apresenta-se o problema em estudo nesta dissertação, o problema de lotes de produção multi-artigo capacitado com tempos de preparação, e uma formulação de programação inteira mista para o problema. Neste capítulo apresentam-se ainda algumas extensões para o problema e um pequeno exemplo para ilustrar a sua formulação. Por último, revêem-se os métodos de resolução quer para o problema de lotes de produção multi-artigo capacitado com tempos de preparação, quer para o problema sem tempos de preparação.

No capítulo 3 revêem-se os conceitos teóricos da decomposição de Dantzig-Wolfe, do método de geração de colunas e do método de partição e geração de colunas.

O capítulo 4 é dedicado à aplicação de duas decomposições de Dantzig-Wolfe ao problema LPMACC e à resolução dos problemas reformulados pelo método de partição e geração de colunas. Para ambas as decomposições, define-se o problema mestre, os subproblemas e o primeiro problema mestre restrito associados a cada modelo de decomposição, e resolve-se a relaxação linear de ambos pelo método de geração de colunas. Para resolver o problema inteiro, aplica-se o método de partição e geração de colunas, definindo-se regras de partição baseadas nas variáveis originais. Para exemplificar a aplicação destes conceitos ao problema LPMACC, apresenta-se a resolução de um pequeno exemplo.

No capítulo 5 analisam-se os aspectos relacionados com a implementação dos dois modelos de decomposição e apresentam-se alguns testes computacionais. Nos testes computacionais, comparam-se os resultados obtidos com a decomposição por artigo e com a decomposição por período, com os resultados obtidos usando um "solver" de problemas de programação inteira mista, o CPLEX (ILOG 2002).

Finalmente no capítulo 6, apresentam-se as conclusões gerais deste trabalho e estabelecem-se algumas sugestões de trabalho futuro.

2 O Problema de Lotes de Produção Multi-Artigo Capacitado com Tempos de Preparação

Este capítulo é dedicado à apresentação do problema em estudo nesta dissertação: o problema de lotes de produção multi-artigo capacitado com tempos de preparação.

Na secção 2.1 faz-se um enquadramento geral do sistema de planeamento e controlo de produção e revêem-se os conceitos gerais deste sistema. Com esta secção, pretende-se estabelecer o paralelo entre o problema LPMACC e o plano mestre de produção do sistema de planeamento e controlo da produção e enquadrar o problema LPMACC no âmbito do planeamento da produção.

Na secção 2.2 revêem-se e classificam-se alguns dos modelos de lotes de produção que mais têm sido estudados na literatura e apresentam-se as principais contribuições. De referir que, dada a enorme variedade de modelos, esta revisão de bibliografia é centrada nos modelos de nível único, com procura determinística e dinâmica, pois o LPMACC insere-se nesta classe de problemas.

A secção 2.3 é dedicada à apresentação do problema de lotes de produção multi-artigo capacitado com tempos de preparação. Na subsecção 2.3.1 define-se o problema, na subsecção 2.3.2 apresenta-se uma formulação de programação inteira mista para o problema, que se designa de formulação clássica, na subsecção 2.3.3 apresentam-se algumas das extensões mais comuns para o problema e por último, na subsecção 2.3.4, apresenta-se um pequeno exemplo para ilustrar a formulação clássica.

Por sua vez na secção 2.4 revêem-se os métodos de resolução, quer do problema de lotes de produção multi-artigo capacitado com tempos de preparação, quer do problema sem tempos de preparação. A revisão é feita com base na divisão dos métodos de resolução em três categorias: métodos exactos, heurísticas de senso comum e heurísticas baseadas em modelos de programação matemática.

2.1 Planeamento e Controlo da Produção

Segundo (Vollmann, Berry et al. 1997): "Um sistema de planeamento e controlo da produção pode ser visto como uma ferramenta de apoio à tomada de decisão em ambientes industriais, que fornece aos gestores informações adequadas para a gestão eficiente do fluxo de materiais, para a utilização eficiente de pessoas e equipamentos, para a coordenação das actividades internas com as actividades dos fornecedores e para a comunicação com os clientes sobre as necessidades do mercado".

O processo de produção envolve a transformação de matérias primas em produtos acabados, habitualmente através da elaboração de uma sequência de passos que podem produzir e/ou consumir produtos intermédios. O planeamento da produção encarrega-se de planear os recursos e actividades necessárias para que o processo de produção seja cumprido de uma forma eficiente. Na Figura 1 apresenta-se simplificadamente o sistema de planeamento e controlo da produção, que se passa a detalhar.

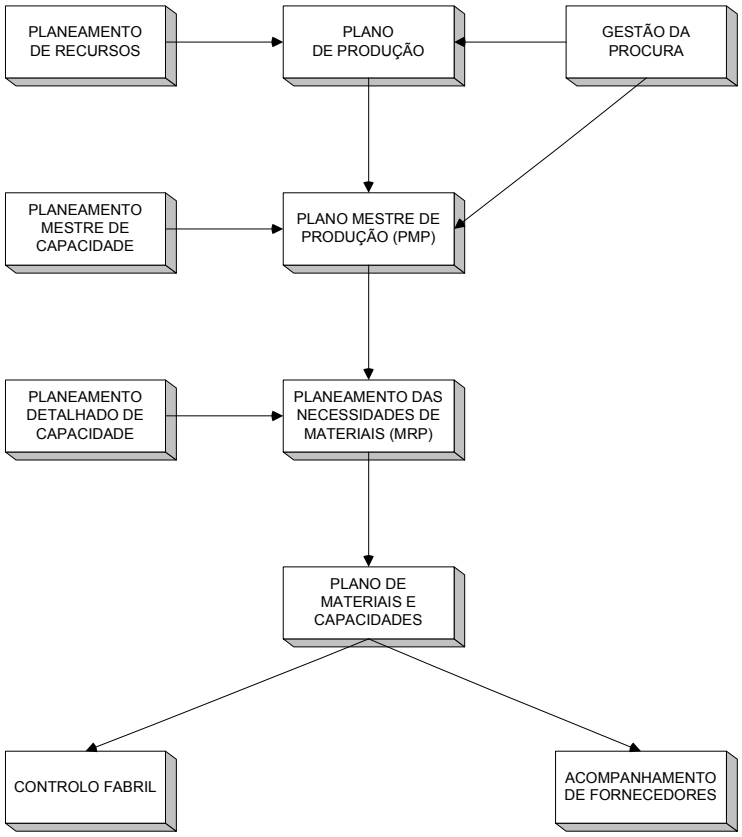

Figura 1: Sistema de planeamento e controlo da produção.

O planeamento é feito a três níveis: no longo-prazo, no médio-prazo e no curto-prazo. No primeiro nível de planeamento estabelecem-se as decisões estratégicas da empresa, através da

definição de objectivos para o planeamento e controlo da produção. Da informação recolhida do planeamento de recursos e da gestão da procura, resulta um plano de produção. Nesse plano de produção estabelecem-se os objectivos globais da empresa em termos de produção no longo prazo (habitualmente para um horizonte de planeamento de cerca de um ano), definindo-se as quantidades a produzir em termos agregados em cada período (normalmente em termos de famílias de produtos, pois apenas são conhecidas as previsões da procura em termos agregados e não em termos de produtos) e as necessidades agregadas de capacidade.

O planeamento a médio-prazo é mais exacto que o anterior nível de planeamento. Aqui, as incertezas provocadas pelas previsões são gradualmente substituídas por encomendas. Neste nível elaboram-se de uma forma faseada o plano mestre de produção (PMP) e o plano das necessidades de materiais e capacidades.

O plano mestre de produção resulta da desagregação do plano de produção e neste plano procura-se determinar que artigos específicos devem ser produzidos nos próximos períodos de planeamento, em que quantidades e para que datas. Em cada período estabelecem-se portanto as quantidades necessárias em termos de produtos acabados.

O plano mestre de produção serve de "input" ao planeamento das necessidades de materiais (MRP). Com base nas quantidades e datas programadas dos produtos acabados especificados no PMP, no planeamento das necessidades de materiais determinam-se os componentes necessários (fazendo a explosão de materiais), em que quantidades são necessários e em que datas deve iniciar-se a sua produção/compra, para que o plano mestre de produção definido seja cumprido.

Quer o PMP, quer o MRP são sujeitos a um processo de validação através de duas técnicas de planeamento da capacidade. Estas técnicas são respectivamente: o planeamento mestre da capacidade e o planeamento detalhado da capacidade. Ambas as técnicas verificam a viabilidade e a exequibilidade dos planos definidos. Sempre que a carga (ou trabalho planeado) ultrapassa a capacidade disponível o planeamento da capacidade invalida a execução dos planos e acções correctivas são sugeridas. Estas acções sugerem alterações ao nível da carga (ajuste dos perfis de carga, negociação/atrasos de datas de entrega, entregas parciais, entre outras) ou ao nível da capacidade (recurso a horas extraordinárias, subcontratar, etc.). Da validação do MRP resulta o plano de materiais e capacidades, que será usado no planeamento a curto-prazo.

O planeamento a curto-prazo é realizado após a conclusão do planeamento das necessidades de materiais e respectiva validação pelo planeamento da capacidade. A este nível a

palavra planeamento dá lugar à palavra programação.

As actividades normalmente envolvidas na programação da produção são a definição e controlo de prioridades atendendo à carga de trabalhos e capacidade disponível, o sequenciamento de ordens de fabrico, a afectação de trabalhadores às estações de trabalho, entre outras. São estas actividades do dia-a-dia, que dentro da capacidade disponível permitem o cumprimento das ordens de fabrico. A programação da produção reveste-se de grande complexidade, atendendo às variáveis envolvidas que são altamente dinâmicas e à imperiosa necessidade de informação actual e válida. Neste contexto, um sistema de informação capaz de disponibilizar informação útil à avaliação de situações e tomada de decisões torna-se indispensável.

Em suma pode dizer-se que, num sistema de planeamento e controlo da produção, os planos de produção iniciados a longo-prazo são gradualmente detalhados até à especificação das tarefas diárias e integrados num sistema de gestão global, com vista à eficiente utilização dos recursos de modo a garantir o fabrico do "mix" de artigos pedidos, respeitando datas de entrega pré-determinadas.

2.2 Lotes de Produção

O problema da determinação de lotes de produção, está intimamente ligado ao plano mestre de produção do sistema de planeamento e controlo da produção.

Existe uma enorme variedade de modelos/problemas de lotes de produção. O problema da determinação de lotes de produção é um problema central do planeamento da produção, que se relaciona com a tomada de decisão em termos do tamanho dos lotes de um ou vários artigos que têm de ser processados e em termos do instante de tempo em que estes artigos devem ser processados. As decisões de produção são habitualmente tomadas atendendo ao melhor balanço entre os objectivos financeiros e os objectivos de satisfação do serviço ao cliente.

Estes problemas surgem essencialmente em ambientes industriais e têm sido amplamente estudados na literatura ao longo das últimas décadas. Nesta secção apresenta-se uma classificação dos vários tipos de modelos de lotes de produção de nível único, com procura determinística e dinâmica que têm sido estudados e revêem-se algumas das contribuições mais importantes para esses modelos. Esta revisão de bibliografia é centrada apenas nos modelos de nível único, com procura determinística e dinâmica, visto o problema em estudo nesta

dissertação, estar inserido nesta classe de problemas. Para uma visão mais detalhada sobre os modelos de lotes de produção, recomenda-se a leitura das seguintes referências: (Rizk and Martel 2001) e (Kuik, Salomon et al. 1994)

Nos problemas de lotes de produção de nível único, a estrutura do(s) artigo(s) (frequentemente referida como lista de materiais) é constituída por apenas um nível, significando que os artigos passam directamente de matéria prima a produto acabado quando são processados, não existindo artigos intermédios (semi acabados) ou postos de montagem. Visto de uma outra perspectiva, pode-se dizer que estamos perante um problema de nível único caso a procura de cada artigo seja independente da procura dos restantes.

Quanto ao tipo de procura, foi já referido que os modelos que se abordam nesta secção têm procura determinística e dinâmica. Para que um problema tenha procura determinística, o valor da procura futura tem de ser conhecido. Se esse valor não for conhecido com exactidão e for baseado em probabilidades, a procura passa a ser estocástica (ou probabilística). A procura, pode ainda ser dinâmica ou estática. Estamos perante um problema com procura dinâmica, quando esta varia ao longo do tempo e perante um problema com procura estática, quando esta se mantém constante ao longo do tempo.

Embora estes problemas de nível único não descrevam a situação real de muitos sistemas de produção, têm atraído tal como já referido a atenção de um amplo número de investigadores. Isto deve-se ao facto destes poderem ser vistos como rotinas dos modelos de multi-nível, e poderem ser generalizados para lidar com situações de multi-nível, para além de serem modelos cujo conhecimento pode auxiliar na resolução dos modelos mais complexos.

Na Tabela 1, apresenta-se uma classificação dos modelos de lotes de produção de nível único com procura determinística e dinâmica, segundo duas características. A primeira característica está relacionada com a existência ou não de limitações de capacidade e a segunda com a consideração de um ou vários artigos. Desta divisão por características, resultam quatro tipos de modelos. Para cada tipo de modelo, apresentam-se os problemas que têm sido estudados na literatura e citam-se alguns dos autores que contribuíram para o seu desenvolvimento e resolução.

	Problema não capacitado	Problema capacitado
Artigo único	- Uncapacitated lot-sizing problem: (Wagner and Whitin 1958); (Zangwill 1969).	- Capacitated lot-sizing problem: (Florian and Klein 1971); (Florian, Lenstra et al. 1980). - Continuous setup lot-sizing problem: (Florian, Lenstra et al. 1980); (Karmarkar, Kekre et al. 1987); (Constantino 1996). - Discrete lot-sizing and scheduling problem: (Salomon 1991); (Hoesel and Kolen 1994).
Multi-artigo		- Capacitated lot-sizing problem: (Manne 1958); (Dzielinski and Gomory 1965); (Lasdon and Terjung 1971); (Eppen and Martin 1987); (Trigeiro, Thomas et al. 1989); (Chen and Thizy 1990); (Diaby, Bahl et al. 1992a); (Diaby, Bahl et al. 1992b); (Miller, Nemhauser et al. 2000); (Jans 2002). - Continuous setup lot-sizing problem: (Karmarkar and Schrage 1985);(Pochet and Wolsey 1991); (Constantino 1996). - Discrete lot-sizing and scheduling problem: (Fleischmann 1990); (Salomon 1991); (Hoesel and Kolen 1994); (Miller and Wolsey 2001).

Tabela 1: Modelos de nível único com procura determinística e dinâmica.

O problema de lotes de produção de artigo único não capacitado esteve na origem dos primeiros desenvolvimentos no campo dos problemas de lotes de produção. O trabalho pioneiro na área dos lotes de produção deveu-se a Harris (Harris 1913) e deu origem a um modelo amplamente conhecido, o modelo da quantidade económica de encomenda ("EOQ – Economic Order Quantity"). Wagner e Whitin (Wagner and Whitin 1958) apresentaram para o problema de lotes de produção de artigo único não capacitado, conhecido na literatura por "ULS –

Uncapacitated Lot-sizing Problem", uma formulação de programação matemática e um algoritmo de programação dinâmica, que resolve o problema até à optimalidade, que ficou conhecido como algoritmo de Wagner-Whitin. Em termos de revisão de bibliografia de lotes de produção é habitual associarem-se estes três autores aos primeiros desenvolvimentos neste campo de investigação.

O problema de lotes de produção multi-artigo não capacitado pode ser visto como um conjunto de problemas independentes de lotes de produção de artigo único não capacitado, pelo que o tipo de problema que assenta nesta classe acaba por se converter num conjunto de problemas de artigo único não capacitado.

Nos problemas de lotes de produção multi-artigo capacitados é habitual estabelecer-se a distinção entre *"small bucket problems "* e *"big bucket problems"*.

Nos *"big bucket problems"* permite-se a produção de vários artigos em cada máquina durante um período de planeamento. Usando esta terminologia, pode dizer-se que o problema de lotes de produção multi-artigo capacitado, conhecido na literatura por "CLSP – Capacitated Lot-sizing Problem", é um *"big bucket problem"*. Este problema baseia-se na determinação de um plano de produção faseado no tempo, que indica a quantidade a produzir de um conjunto de artigos e quando é que essa quantidade deve ser produzida. O plano de produção a determinar deverá tomar em linha de conta todas as condicionantes ao problema, por forma a garantir uma utilização eficiente dos recursos de produção disponíveis e garantir a satisfação das ordens de compra dos clientes, nas quantidades devidas e no prazo de entrega acordado, com vista à satisfação do serviço ao cliente. Para além destas condicionantes, o plano de produção deverá reflectir a minimização de todos os custos a ele associados.

Nos *"small bucket problems"* considera-se que em cada período de planeamento apenas um artigo pode ser produzido em cada máquina. Neste tipo de problemas, é habitual fazer-se ainda uma distinção adicional entre o problema "CSLP – Continuous Set Up Lot-sizing Problem" e o "DLSP – Discrete Lot-sizing and Scheduling Problem". Enquanto que no CSLP, a produção realizada em cada período pode variar, mas é limitada pela restrição de capacidade, no DLSP, a quantidade produzida em cada período ou é nula, ou tem de ser igual à capacidade total desse período.

Os problemas CLSP, CSLP e DLSP para o caso de multi-artigo são uma extensão dos mesmos problemas para o caso de artigo único. Em termos de definição, estes três problemas estão intimamente relacionados, pois em todos eles existe o objectivo de definir planos de

produção que minimizem todos os custos relacionados e que, por outro lado, permitam a verificação de um conjunto de restrições. O tipo de plano de produção associado a cada um dos três problemas é contudo diferente, devido às particularidades de cada um, já acima apresentadas.

2.3 O problema de Lotes de Produção Multi-Artigo Capacitado com Tempos de Preparação

O problema abordado nesta dissertação assenta na classe de problemas de lotes de produção de nível único, multi-artigo, com procura determinística e dinâmica e capacitado, é um *"big bucket problem"* e corresponde ao problema resolvido ao nível do plano mestre de produção, no sistema de planeamento e controlo da produção.

O problema de lotes de produção multi-artigo capacitado é um dos problemas de lotes de produção que mais tem sido estudado ao longo dos últimos anos. Este interesse particular deve-se ao facto de este ser um problema importante ao nível do planeamento da produção, que como se sabe tem sido uma área na qual se têm verificado elevados progressos no desenvolvimento e aplicação de ferramentas, que auxiliem as organizações na tomada de decisão.

O problema LPMACC, tal como já referido, permite a produção de vários lotes de artigos diferentes na mesma máquina durante um único período e é um problema de nível único uma vez que nenhum dos artigos pode ser predecessor de outro artigo (ou seja nenhum artigo poderá servir como componente de um outro artigo). Em cada período e para cada artigo é conhecida uma procura determinística e dinâmica, pois considera-se que a procura futura é conhecida e que pode variar ao longo dos períodos. Em cada período existe uma capacidade limitada, que interliga os diferentes artigos a produzir, pelo facto de os recursos utilizados na sua produção serem os mesmos.

Na versão mais estudada do problema de lotes de produção multi-artigo capacitado, na sua formulação, apenas se consideram os custos de preparação (ou de setup), sendo a preparação contabilizada como um custo de preparação na função objectivo. Como o tempo de preparação consome parte da capacidade de produção disponível, nesta versão do problema não existe a garantia de que a solução encontrada para o problema seja a verdadeira solução óptima, e que seja sequer admissível. Para muitos dos autores que consideram na formulação do problema apenas o custo de preparação, a inclusão deste tempo no modelo é vista como uma extensão simples ao modelo. Esta posição foi contrariada por (Trigeiro, Thomas et al. 1989), que mostrou que o problema com a consideração dos tempos de preparação se torna num

problema muito mais difícil de resolver. Com vista à modelação mais precisa do problema, na versão em estudo nesta dissertação considera-se directamente no modelo a capacidade gasta com a preparação, ao invés de esta ser considerada implicitamente no custo de preparação. Para distinguir o problema que considera só os custos de preparação do que considera os custos e tempos de preparação, o primeiro passará doravante a designar-se por LPMACS e o segundo por LPMACC.

2.3.1 Definição do Problema

O problema LPMACC é um problema que surge no planeamento da produção. Em linhas gerais, neste problema procura-se determinar um plano de produção para um conjunto de artigos, faseado no tempo, através da consideração de um conjunto de períodos de planeamento. Esse plano indica para cada artigo as quantidades a produzir desse artigo (que definem um lote) e os períodos em que essas quantidades devem ser produzidas, atendendo a um conjunto de restrições.

Cada problema pode ter vários planos de produção associados, e qualquer plano de produção desse problema engloba um custo de produção, um custo de preparação e um custo de armazenagem. O objectivo do problema passa portanto pela determinação do plano de produção que tem o custo mais baixo.

Por cada unidade produzida de um lote, incorre-se num custo de produção, que pode variar por artigo e por período. O custo de preparação representa um custo fixo do lote, e por último, o custo de armazenagem representa o custo que se incorre por unidade armazenada e por período de armazenagem, sempre que a decisão é a de armazenar artigos para períodos posteriores.

Os custos de preparação e de armazenagem relacionam-se entre si, uma vez que a minimização de um deles, provocará inevitavelmente o aumento do outro. Se se considerar por exemplo um plano de produção no qual se produzem artigos em períodos anteriores e se armazenam estes artigos para satisfazer procuras de períodos posteriores, está-se a minimizar o custo de preparação e a maximizar o custo de armazenagem. Se, por outro lado, se produzir cada artigo no período em que é necessário para satisfazer a sua procura, está-se a minimizar o custo de armazenagem, e a maximizar o custo de preparação. Deverá portanto existir um balanço entre estas duas componentes do custo, que permita a minimização do custo global.

Em cada período e para cada artigo, é conhecida uma procura, que deve ser satisfeita e em cada período existe uma capacidade disponível que deve ser respeitada. O problema em estudo nesta dissertação considera a existência de um custo de preparação e de um tempo de preparação (de máquinas, pessoas, materiais, etc.), associados à decisão de produção, pelo que sempre que se decidir produzir um determinado artigo, antes de se iniciar a produção desse artigo num determinado período, deve ocorrer uma preparação e nesta preparação devem considerar-se o tempo de preparação e o respectivo custo de preparação. O tempo de preparação pode variar por artigo e por período e consome parte da capacidade de produção disponível e a produção do lote só se poderá iniciar após a preparação estar completa.

2.3.2 Formulação Clássica

Nesta subsecção apresenta-se uma formulação de programação inteira mista para o problema LPMACC, que se designa por formulação clássica.

Na formulação clássica consideram-se três tipos de variáveis de decisão. Um conjunto de variáveis que podem tomar qualquer valor não negativo, que representam as quantidades a produzir de cada artigo em cada período; outro conjunto de variáveis binárias, que assumem o valor *1*, sempre que para um determinado artigo, num determinado período, a decisão seja a de produzir, e que assumem o valor *0*, caso contrário; e por último um conjunto de variáveis que podem assumir qualquer valor não negativo, que representam o nível de inventário de um determinado artigo, no final de um determinado período.

Existem quatro conjuntos de restrições ao problema. O primeiro conjunto representa as restrições de equilíbrio de stocks, que garantem a satisfação da procura de cada artigo em cada período; o segundo conjunto representa as restrições de capacidade, nas quais se toma em linha de conta a capacidade disponível em cada período; e o terceiro conjunto, que garante a ocorrência de uma preparação, sempre que a decisão seja a de produzir um determinado artigo num determinado período. Existe ainda um quarto conjunto de restrições nas quais se força, para cada artigo, o inventário no final do último período de planeamento a ser nulo.

Na função objectivo do problema minimizam-se os custos totais de produção, de preparação e de inventário.

Uma solução válida para o problema corresponde às quantidades a produzir de cada artigo em cada período e às quantidades a armazenar de cada artigo para períodos posteriores,

sem ser ultrapassada a capacidade disponível em cada período, garantindo a entrega atempada dos artigos e garantindo a ocorrência de uma preparação sempre que se produza. Uma solução óptima corresponderá a uma solução cujo custo global é inferior ou igual ao custo de qualquer solução válida.

Nos parágrafos seguintes, apresenta-se uma formulação clássica para o problema LPMACC, e explica-se detalhadamente essa formulação.

Considerem-se os seguintes conjuntos, parâmetros e variáveis de decisão:

Conjuntos:

I representa o conjunto de artigos, indexados por $i=1,...,n$

J representa o conjunto de períodos de produção, indexados por $j=1,...,m$

Parâmetros:

p^i_j representa o custo unitário de produção do artigo *i* no período $j;\ i=1,...,n\ ;\ j=1,...,m$

q^i_j representa o custo fixo de preparação do artigo *i* no período $j;\ i=1,...,n\ ;\ j= 1,...,m$

h^i_j representa o custo unitário de armazenagem do artigo *i* no final do período $j;\ i=1,...,n\ ;\ j=1,...,m$

d^i_j representa a procura do artigo *i* no período $j;\ i=1,...,n\ ;\ j=1,...,m$

c_j representa a capacidade de produção no período $j;\ j=1,...,m$

a^i_j representa a capacidade consumida na produção de uma unidade do artigo *i* no período $j;\ i=1,...,n\ ;\ j=1,...,m$

b^i_j representa a capacidade consumida na preparação do artigo *i* no período $j;\ i=1,...,n\ ;\ j=1,...,m$

Variáveis de Decisão:

x^i_j representa a quantidade a produzir do artigo *i* no período $j;\ i=1,...,n\ ;\ j=1,...,m$

y^i_j é uma variável binária que assume o valor *1* se o artigo *i* é produzido no período *j*, e assume o valor *0* caso contrário; $i=1,...,n\ ;\ j=1,...,m$

Variáveis de Decisão Auxiliares:

s^i_j representa o nível de inventário do artigo *i* no final do período $j;\ i=1,...,n\ ;\ j=1,...,m$

A formulação clássica do modelo de lotes de produção multi-artigo capacitado com tempos de preparação, considerando *I* artigos e *J* períodos, é a que se apresenta de seguida:

$$Z_C = Min \sum_{i=1}^{n}\sum_{j=1}^{m} p_j^i x_j^i + \sum_{i=1}^{n}\sum_{j=1}^{m} q_j^i y_j^i + \sum_{i=1}^{n}\sum_{j=1}^{m} h_j^i s_j^i \qquad (C)$$

sujeito a:

$$x_1^i = d_1^i + s_1^i, \quad \forall i \in I \qquad (2.1)$$

$$s_{j-1}^i + x_j^i = d_j^i + s_j^i, \quad \forall i \in I, \forall j \in J \setminus \{1\} \qquad (2.2)$$

$$\sum_{i=1}^{n} a_j^i x_j^i + \sum_{i=1}^{n} b_j^i y_j^i \le c_j, \quad \forall j \in J \qquad (2.3)$$

$$x_j^i \le min\left\{\frac{c_j - b_j^i}{a_j^i}, \sum_{t=j}^{m} d_t^i\right\} y_j^i, \quad \forall i \in I, \forall j \in J \qquad (2.4)$$

$$s_m^i = 0, \quad \forall i \in I \qquad (2.5)$$

$$s_j^i \ge 0, \quad \forall i \in I, \forall j \in J \setminus \{m\} \qquad (2.6)$$

$$x_j^i \ge 0, \quad \forall i \in I, \forall j \in J \qquad (2.7)$$

$$y_j^i \in \{0,1\}, \quad \forall i \in I, \forall j \in J\,. \qquad (2.8)$$

A função objectivo minimiza a soma do custo total. As restrições *(2.1)* e *(2.2)* são as restrições de equilíbrio de stocks, que garantem que a procura dos clientes em cada período é satisfeita e que relacionam os níveis de inventário de um determinado período com os níveis de inventário do período imediatamente anterior. Nestas restrições para um determinado artigo, força-se a que a soma da procura num determinado período com o nível de inventário nesse período seja igual ao nível de inventário do período imediatamente anterior a esse período mais a quantidade produzida nesse período. Na Figura 2, ilustra-se esquematicamente a restrição de equilíbrio de stocks, relativa ao artigo *i* no período *j*.

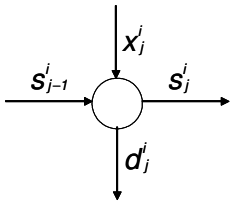

Figura 2: Restrição de equilíbrio de stocks.

Em *(2.3)* tem-se as restrições de capacidade. Estas restrições forçam o consumo do recurso com a produção e a preparação a não exceder a capacidade disponível no período *j*. As

restrições *(2.4)* e *(2.8)*, garantem que uma preparação ocorre se o artigo *i* for produzido no período *j*. As restrições *(2.4)* poderiam ser substituídas por $x^i_j \leq M y^i_j$, sendo *M* um valor constante elevado. Deste modo garantir-se-ia que, sempre que se decidisse produzir o artigo *i* no período *j*, deveria ocorrer uma preparação. Para tornar a formulação mais forte, o coeficiente *M* pode ser substituído por um lado por $\frac{c_j - b^i_j}{a^i_j}$, ou seja, pela quantidade máxima que é possível produzir do artigo *i* no período *j*, e por outro lado por $\sum_{t=j}^{m} d^i_t$, ou seja, pela soma da procura desde o período em análise (período *j*) até ao último período de planeamento. Entre os dois coeficientes referidos, deve escolher-se aquele que tiver menor valor, uma vez que a quantidade a produzir do artigo *i* no período *j* será limitada por esse valor. As restrições *(2.6)* e *(2.7)* impõem a condição de não negatividade das variáveis.

Por último, as restrições *(2.5)* forçam o inventário final de cada artigo a ser nulo. Estas restrições são redundantes, pois como na função objectivo estamos a minimizar custos, nunca compensa no último período de planeamento armazenar unidades para períodos posteriores. Considerar o inventário final nulo não retira generalidade ao modelo relativamente ao modelo que admite a possibilidade de existência de inventário final, uma vez que é possível anular o nível desse inventário, adicionando-o à procura do último período. Como resultado, obtém-se $s^i_m = 0$ e d'^i_m, sendo d'^i_m a soma da procura do último período com o nível de inventário final.

No primeiro período de planeamento, caso a capacidade disponível não seja suficiente para fazer face à procura deste período, a solução para o problema será impossível, pelo facto de não se considerar a possibilidade de existência/compra de stock no início do primeiro período de planeamento. Em termos do modelo matemático, por forma a assegurar a existência de uma solução admissível, introduz-se nas restrições *(2.1)* uma variável artificial para cada artigo (variáveis v^i), à qual está associado um custo artificial f^i, que representa uma penalidade elevada na função objectivo, garantindo-se deste modo um valor mínimo de v^i. Em termos de alterações ao modelo, na função objectivo passa a ter de se considerar o seguinte termo: $\sum_{i=1}^{n} f^i v^i$ e as restrições *(2.1)* passam a ser definidas por:

$$v^i + x^i_1 = d^i_1 + s^i_1 \quad \forall i \in I$$

Um novo conjunto de restrições deve ser adicionado, para expressar a condição de não negatividade das variáveis v^i.

2.3.3 Extensões para o LPMACC

Nesta subsecção, apresentam-se algumas das extensões mais comuns para o problema LPMACC.

Uma das variações ao problema surge caso se considere a possibilidade de entregas com atraso. Neste caso, a entrega da procura que está agendada para um determinado período é adiada até um determinado número de períodos mais tarde. Em termos do modelo as variáveis s_j^i passam a não ter restrição de sinal, assumindo-se que os clientes estão dispostos a receber os artigos com um atraso relativamente à data pré-determinada no plano, desde que esse atraso não ultrapasse um determinado número de períodos.

Uma outra extensão ao problema está relacionada com o planeamento de recursos. Nesta extensão, os níveis de recursos disponíveis podem ser ajustados ao longo do horizonte de planeamento, consoante as necessidades de procura. Uma possibilidade é o recurso a trabalho extraordinário, adaptando-se desta forma a capacidade disponível em cada período.

Por último, refere-se uma extensão relativa a limites na produção. Neste caso considera-se que poderão existir limites inferiores ou superiores, em termos das quantidades a produzir de cada artigo em cada período.

2.3.4 Exemplo

Nesta subsecção, apresenta-se um pequeno exemplo para ilustrar a formulação clássica do problema LPMACC, recorrendo-se a uma instância de um problema com 2 artigos e 3 períodos de planeamento, cujos recursos de produção estão expressos em unidades de tempo. Na Tabela 2 e na Tabela 3, apresentam-se os parâmetros da instância, sendo de referir que, neste caso particular, os custos de produção, de preparação e de armazenagem não variam com o tempo, assim como o tempo de produção e o tempo de preparação.

	Artigo 1	Artigo 2
$\mathbf{p^i}$ (unidades monetárias/unidade)	1	1
$\mathbf{a^i}$ (unidades de tempo/unidade)	1	1
$\mathbf{q^i}$ (unidades monetárias)	200	200
$\mathbf{b^i}$ (unidades de tempo)	10	20
$\mathbf{h^i}$ (unidades monetárias/unidade)	1	1
$\mathbf{f^i}$ (unidades monetárias)	2580	2580

Tabela 2: Parâmetros do Exemplo.

	Período 1	Período 2	Período 3
c_j (unidades de tempo)	250	250	250
d^1_j (unidades)	100	80	50
d^2_j (unidades)	100	50	80

Tabela 3: Parâmetros do Exemplo.

Na Figura 3 e na Figura 4, recorre-se à representação em rede, para exemplificar as restrições de equilíbrio de stocks (restrições *(2.1)* e *(2.2)*). Se na formulação clássica não se considerarem as restrições de capacidade *(2.3)* obtém-se um problema para cada artigo, que pode ser visto como um fluxo numa rede. Os arcos com fluxo representam arcos que contribuem para a satisfação da procura e ou são arcos de produção, ou são arcos de armazenagem, ou são arcos de entrega. Na Figura 3, representam-se as restrições relativas ao artigo 1 e na Figura 4 as restrições relativas ao artigo 2. Na Figura 3, $\sum_{k=1}^{3} d_k^1$ representa a procura total do artigo 1, assim como na Figura 4 $\sum_{k=1}^{3} d_k^2$ representa a procura total do artigo 2.

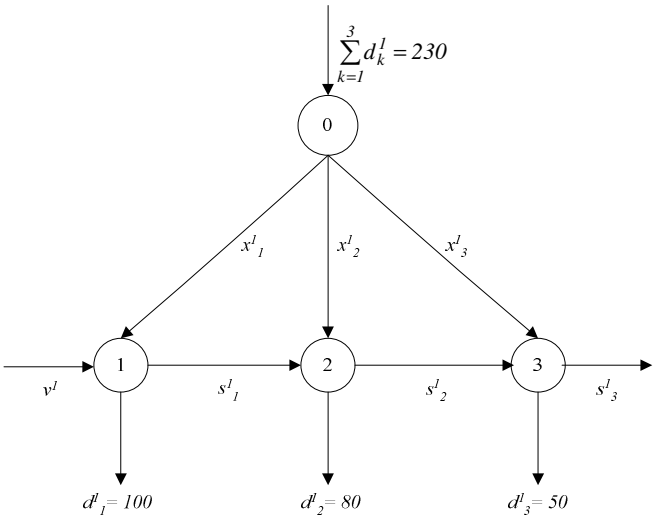

Figura 3: Representação em rede das restrições de equilíbrio de stocks do artigo 1.

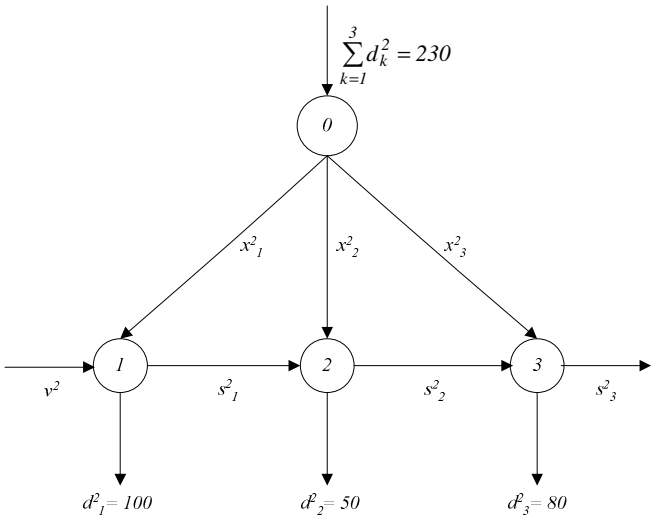

Figura 4: Representação em rede das restrições de equilíbrio de stocks do artigo 2.

Na Figura 5 apresenta-se matricialmente a formulação clássica do Exemplo. Refere-se que o primeiro índice das variáveis diz respeito ao artigo e o segundo índice ao período.

	v1	**v2**	**x11**	**x12**	**x13**	**y11**	**y12**	**y13**	**s11**	**s12**	**s13**	**x21**	**x22**	**x23**	**y21**	**y22**	**y23**	**s21**	**s22**	**s23**		
i=1, j=1	1		1						-1												=	100
i=1, j=2				1					1	-1											=	80
i=1, j=3					1					1	-1										=	50
i=2, j=1		1										1						-1			=	100
i=2, j=2													1					1	-1		=	50
i=2, j=3														1					1	-1	=	80
j=1			1			10						1			20						≤	250
j=2				1			10						1			20					≤	250
j=3					1			10						1			20				≤	250
i=1, j=1			1			-230															≤	0
i=1, j=2				1			-130														≤	0
i=1, j=3					1			-50													≤	0
i=2, j=1												1			-230						≤	0
i=2, j=2													1			-130					≤	0
i=2, j=3														1			-80				≤	0
custo	2580	2580	1	1	1	200	200	200	1	1	1	1	1	1	200	200	200	1	1	1		

Figura 5: Representação matricial da formulação clássica - Exemplo.

O custo total da solução óptima da instância apresentada é de 1510 unidades monetárias. A solução correspondente apresenta-se na Figura 6 e na Figura 7. Todos os arcos nos quais existe fluxo estão representados a tracejado. Na Figura 6 representa-se o plano óptimo de produção do artigo 1 e na Figura 7 o plano óptimo do artigo 2. Como as variáveis de preparação não estão representadas nas redes, apresentam-se seguidamente os seus valores: para o artigo 1, na solução óptima $y^1_1 = y^1_2 = 1$ e $y^1_3 = 0$; para o artigo 2, $y^2_1 = y^2_2 = y^2_3 = 1$.

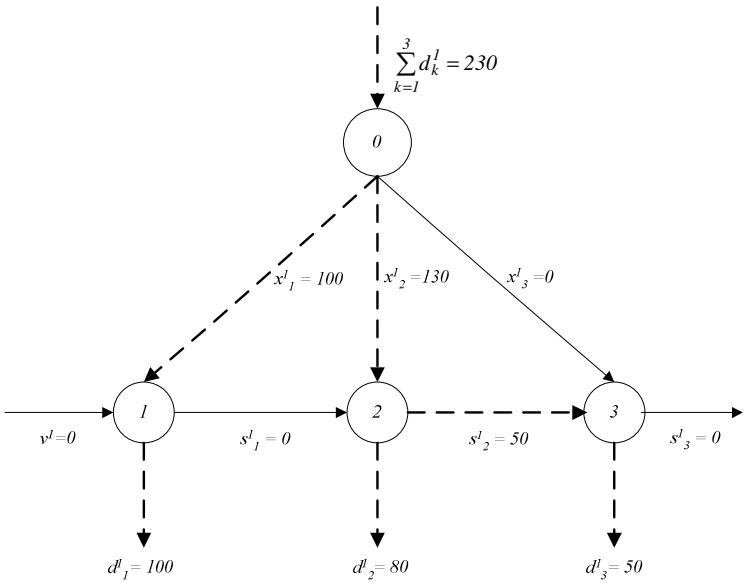

Figura 6: Representação em rede do plano óptimo de produção do artigo 1.

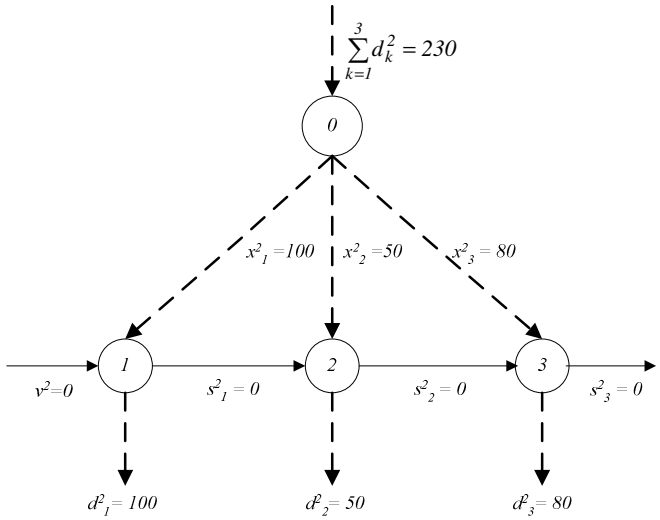

Figura 7: Representação em rede do plano óptimo de produção do artigo 2.

2.4 Métodos de Resolução

Os problemas de lotes de produção capacitados têm sido estudados por vários autores ao longo do tempo, devido à sua complexidade e importância ao nível do planeamento da produção. Nas últimas décadas têm sido publicados na literatura vários trabalhos que apresentam diferentes métodos de resolução, quer para o problema de lotes de produção multi-artigo capacitado sem tempos de preparação (LPMACS), quer para o problema com tempos de preparação (LPMACC).

Em termos da complexidade do problema, (Florian, Lenstra et al. 1980) e (Bitran and Yanasse 1982) mostraram que o problema de lotes de produção de artigo único capacitado é um problema NP-difícil. Mais tarde (Chen and Thizy 1990) mostraram que o problema LPMACS é também NP-difícil. Em (Maes, McClain et al. 1991) demonstra-se que a determinação de uma solução válida para o problema LPMACC é um problema NP-completo.

De seguida, apresentam-se algumas das contribuições mais importantes para a resolução dos problemas de lotes de produção multi-artigo capacitados. De acordo com a literatura sobre os métodos de resolução utilizados na resolução quer do LPMACS, quer do LPMACC, estes podem classificar-se em três categorias: métodos exactos, heurísticas de senso comum e heurísticas baseadas em modelos de programação matemática.

A principal diferença entre as heurísticas e os métodos exactos reside no facto de, nas heurísticas, não existir a garantia de que a solução óptima do problema seja encontrada, enquanto que os métodos exactos visam a obtenção da sua solução óptima.

Nas heurísticas de senso comum, utilizam-se normalmente regras simples, que se baseiam no conhecimento do sistema. Em termos de tempos de resolução, estas heurísticas de senso comum são em geral mais rápidas que as heurísticas baseadas em modelos de programação matemática, mas a solução obtida com as últimas é normalmente de melhor qualidade.

Seguindo a categorização anteriormente exposta, apresentam-se, na Tabela 4, as principais contribuições para a resolução do problema de lotes de produção multi-artigo capacitado, com e sem tempos de preparação.

Heurísticas de senso comum	(Silver and Meal 1973) (Eisenhut 1975) (Lambrecht and Vanderveken 1979) (Dixon and Silver 1981) (Dogramaci, Panayiotopoulos et al. 1981) (Maes and Wassenhove 1986a) (Maes and Wassenhove 1986b) (Kirka and Kokten 1994)
Heurísticas baseadas em modelos de programação matemática	(Manne 1958) (Dzielinski and Gomory 1965) (Lasdon and Terjung 1971) (Bahl 1983) (Thizy and Wassenhove 1985) (Trigeiro, Thomas et al. 1989) (Chen and Thizy 1990) (Cattrysse, Maes et al. 1990) (Diaby, Bahl et al. 1992b) (Merle, Goffin et al. 1999)
Métodos exactos	(Barany, Roy et al. 1984) (Eppen and Martin 1987) (Pochet and Wolsey 1991) (Diaby, Bahl et al. 1992a) (Armentano, França et al. 1999) (Belvaux and Wolsey 2000) (Miller, Nemhauser et al. 2000) (Jans 2002)

Tabela 4: Métodos de resolução do problema de lotes de produção multi-artigo capacitado.

As duas principais abordagens ao problema, com e sem tempos de preparação, são a relaxação lagrangeana e a decomposição de Dantzig-Wolfe, podendo ambas ser utilizadas no contexto heurístico ou exacto. Na relaxação lagrangeana, destaca-se a utilização do método do subgradiente para resolver o problema reformulado e do método de partição e avaliação, para determinar a sua solução óptima inteira. De referir que, em cada nodo da árvore de pesquisa, a relaxação lagrangeana/método do subgradiente são usados para determinar a solução óptima desse nodo. Na decomposição de Dantzig-Wolfe recorre-se habitualmente ao método de geração de colunas para resolver o problema reformulado e ao método de partição e avaliação para determinar a sua solução óptima inteira. Neste caso, em cada nodo da árvore de pesquisa, utiliza-se a decomposição de Dantzig-Wolfe/método de geração de colunas para determinar a solução óptima desse nodo.

As contribuições em termos de heurísticas baseadas em modelos de programação matemática estão associadas essencialmente à relaxação lagrangeana e à decomposição de Dantzig-Wolfe. Nos métodos exactos as principais contribuições relacionam-se com a utilização da relaxação lagrangeana, da decomposição de Dantzig-Wolfe, de planos de corte e, também,

com a utilização da redefinição de variáveis. Importa esclarecer que os métodos de relaxação e de decomposição que são utilizados nas heurísticas baseadas em modelos de programação matemática são os mesmos que se utilizam nos métodos exactos. Aquilo que os distingue é o facto de, nas heurísticas baseadas em modelos de programação matemática, estes métodos serem combinados com heurísticas, não se forçando a obtenção da solução óptima, enquanto que nos métodos exactos se obtém a solução óptima ou uma solução muito aproximada da óptima, que respeita um determinado critério de tolerância. Consideram-se também na categoria de heurísticas baseadas em modelos de programação matemática os trabalhos que apenas abordam a obtenção de limites inferiores para o problema, e os trabalhos onde de algum modo se faz a truncatura do problema para reduzir o esforço computacional na sua resolução.

Nos parágrafos seguintes, pormenoriza-se a utilização quer das heurísticas baseadas em modelos de programação matemática, quer dos métodos exactos.

Os trabalhos pioneiros com vista à resolução do problema de lotes de produção multi-artigo capacitado são atribuídos a (Manne 1958), a (Dzielinski and Gomory 1965) e a (Lasdon and Terjung 1971). Em (Manne 1958) apresenta-se um modelo aproximado de programação linear para o problema. (Dzielinski and Gomory 1965) aplicam a decomposição de Dantzig-Wolfe ao problema e determinam o limite inferior do problema reformulado pelo método de geração de colunas e (Lasdon and Terjung 1971) reformulam o modelo de programação linear proposto por (Manne 1958) e resolvem esse novo problema pelo método de geração de colunas.

A relaxação/decomposição que mais tem sido estudada na literatura consiste na relaxação/decomposição das restrições de capacidade, obtendo-se deste modo um subproblema de lotes de produção de artigo único não capacitado. Referem-se agora alguns dos autores citados na Tabela 4, que abordam esta relaxação/decomposição.

(Trigeiro, Thomas et al. 1989) aplicam a relaxação lagrangeana ao LPMACC e utilizam o método do subgradiente, e uma heurística de alisamento para o resolver. (Chen and Thizy 1990) apresentam para o problema LPMACS, várias relaxações e decomposições. Nesse mesmo artigo, os autores aplicam ao problema a relaxação lagrangeana/método de subgradiente e a decomposição de Dantzig-Wolfe/método de geração de colunas, e estabelecem comparações entre ambos em termos dos limites inferiores obtidos ao resolver os problemas reformulados. (Thizy and Wassenhove 1985) aplicam ao problema LPMACS (e mais tarde (Diaby, Bahl et al. 1992a) ao problema LPMACC) a relaxação lagrangeana, e determinam o limite inferior do problema reformulado através do método do subgradiente. Com base na solução obtida, e fixando as variáveis de preparação, os autores resolvem o problema inteiro heuristicamente

recorrendo a um algoritmo de transportes. (Diaby, Bahl et al. 1992a) resolvem o LPMACC através do método de partição e avaliação ("branch and bound"), utilizando a relaxação lagrangeana e o método do subgradiente, para determinar os limites inferiores. Já (Merle, Goffin et al. 1999) aplicam a relaxação lagrangeana ao problema LPMACC e utilizam um método de planos de corte baseado no centro analítico ("ACCPM-analytic center cutting plane method") para determinar o limite inferior produzido pelo problema reformulado, e comparam este método com o método do subgradiente e o método de geração de colunas. Os mesmos autores utilizam heurísticas primais e duais para obter soluções óptimas inteiras. (Jans 2002) aplica a decomposição de Dantzig-Wolfe e o método de geração de colunas para resolver a relaxação linear do problema LPMACC e aplica o método de partição e geração de colunas para determinar a sua solução óptima inteira. Esta última referência está relacionada com parte do trabalho desenvolvido nesta dissertação. Aquilo que distingue o trabalho de (Jans 2002) de parte do trabalho que aqui se desenvolve é a utilização de regras de partição diferentes no método de partição e geração de colunas.

(Diaby, Bahl et al. 1992a) abordam para o problema LPMACC uma relaxação alternativa, baseada na relaxação das restrições de equilíbrio de stocks, e resolvem o problema reformulado pelo método do subgradiente. Para determinar a solução óptima inteira do problema, os autores aplicam o método de partição e avaliação, e em cada nodo da árvore de pesquisa aplicam a relaxação lagrangeana/método do subgradiente. Este trabalho está de algum modo relacionado com uma parte do trabalho desenvolvido nesta dissertação. Embora aqui se aborde uma decomposição baseada também nas restrições de equilíbrio de stocks, os métodos aplicados na resolução do problema reformulado são diferentes. Nesta dissertação utiliza-se a decomposição de Dantzig-Wolfe/método de geração de colunas e o método de partição e geração de colunas.

Em termos das principais abordagens que utilizam planos de corte são de salientar os trabalhos de: (Barany, Roy et al. 1984) e (Belvaux and Wolsey 2000) que resolvem o problema LPMACS, adicionando ao modelo restrições que correspondem a cortes fortes e resolvem o problema reformulado através do método de partição e avaliação; (Pochet and Wolsey 1991) que utilizam a mesma abordagem no problema LPMACC e de (Miller, Nemhauser et al. 2000) que resolvem o problema LPMACC através do método de partição e corte ("branch-and-cut").

(Eppen and Martin 1987) usam a técnica de redefinição de variáveis (Martin 1987) para resolver o problema LPMACS. O problema reformulado baseia-se numa representação em rede, e apresenta mais variáveis e restrições que a formulação clássica. Contudo o limite inferior deste novo problema é de melhor qualidade que o obtido com a formulação clássica. De realçar que este limite inferior é igual ao que se obtém relaxando ou decompondo as restrições de

capacidade da formulação clássica. Para determinar a solução óptima inteira, os autores utilizam o método de partição e avaliação.

(Armentano, França et al. 1999) representam o problema LPMACC como um problema de fluxo em rede de custo mínimo, e utilizam o método de partição e avaliação para resolver esse problema reformulado.

Nos parágrafos anteriores abordaram-se alguns dos trabalhos mais relevantes em termos da resolução dos problemas de lotes de produção multi-artigo capacitados. Para uma descrição mais pormenorizada sobre os métodos de resolução destes problemas, nomeadamente em termos de heurísticas, aconselha-se a leitura do artigo de (Karimi, Ghomia et al. 2003).

3 Decomposição de Dantzig-Wolfe e Partição e Geração de Colunas

Neste capítulo revêem-se os conceitos teóricos da decomposição de Dantzig-Wolfe (DDW), do método de geração de colunas (MGC) e do método de partição e geração de colunas (MPGC).

Na secção 3.1, introduz-se a decomposição de Dantzig-Wolfe, o método de geração de colunas e a estrutura angular em blocos.

Na secção 3.2 apresentam-se os conceitos teóricos gerais da decomposição de Dantzig-Wolfe e a sua aplicação a um problema de programação linear.

Na secção 3.3 apresentam-se os conceitos teóricos do método de geração de colunas.

Na secção 3.4 aborda-se a aplicação da decomposição de Dantzig-Wolfe a problemas que tenham uma estrutura angular em blocos.

Por último na secção 3.5, expõe-se o método de partição e geração de colunas.

3.1 Introdução

A decomposição de Dantzig-Wolfe aliada à geração de colunas é considerada por muitos autores um caso de sucesso na resolução de problemas de programação linear e de programação inteira de grande dimensão. As origens da DDW remontam ao início da década de 60 (Dantzig and Wolfe 1960), mas o grande "boom" na aplicação deste método verificou-se ao longo da última década.

O método da decomposição de Dantzig-Wolfe (Dantzig and Wolfe 1960) é normalmente utilizado na resolução de problemas de programação linear (PL) ou de programação inteira (PI), cuja dimensão e complexidade limitam a obtenção da solução óptima desses problemas em tempo razoável, e/ou também em problemas de PL e PI que contêm restrições com uma estrutura especial, embora se possa aplicar a qualquer modelo de PL ou PI. A ideia base do método assenta na resolução de um elevado número de problemas de pequena dimensão, tipicamente bem estruturados, em oposição à resolução do problema original, caracterizado por uma complexidade e tamanho que o tornam de difícil, ou até mesmo impossível resolução.

É habitual, em modelos de PL e de PI, existirem vários conjuntos de restrições, que estão associados a apenas algumas variáveis do modelo, que levam à existência de alguns subsistemas de restrições, ligadas por um conjunto distinto de restrições. Estes problemas são conhecidos por apresentarem uma estrutura angular em blocos, com restrições de ligação, tal como se apresenta na Figura 8.

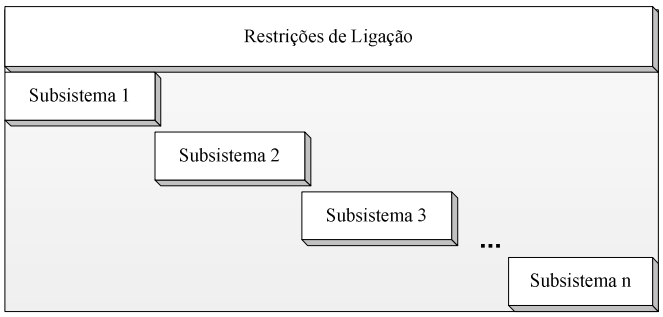

Figura 8: "Layout" de um problema de PL com estrutura angular em blocos.

A existência destes subsistemas faz com que surjam várias submatrizes de zeros no modelo, tal como se pode verificar pela análise da Figura 8. Para tirar partido da estrutura do modelo, na DDW as restrições do problema original são divididas em dois conjuntos. O

primeiro conjunto é definido pelas restrições de ligação, habitualmente constituído pelas restrições difíceis (aquelas que tornam o problema num problema de difícil resolução) e o segundo a partir dos subsistemas. A cada subsistema, está habitualmente associado o mesmo tipo de problema, que exibe uma estrutura especial e para o qual são conhecidos algoritmos de resolução eficientes. Exemplos típicos são o problema de caminho mais curto, o problema de fluxo de custo mínimo e o problema da mochila, entre outros.

Na DDW as restrições de ligação ficam num problema, designado por problema mestre, e define-se um ou mais subproblemas a partir dos subsistemas, existindo partilha de informação entre o problema mestre e os subproblemas, até que a solução óptima para o problema original seja encontrada. Essa solução óptima é obtida através da resolução de uma sequência de problemas de menor dimensão, resolvidos habitualmente pelo método de geração de colunas, desenvolvido na secção 3.3.

Existem inúmeros problemas de optimização, cuja estrutura se adequa à aplicação da DDW. O problema de lotes de produção multi-artigo em estudo nesta dissertação é um exemplo disso, pois pode ser formulado através de modelos com estrutura angular em blocos.

Para um mesmo problema de PL ou PI, são possíveis diferentes decomposições de Dantzig-Wolfe, consoante as restrições que se considerem como pertencendo ao primeiro conjunto e ao segundo. De salientar que os tempos de resolução de um mesmo problema, bem como o seu intervalo de integralidade (ou "gap"), poderão variar significativamente de acordo com a decomposição que se aplique. O intervalo de integralidade, num problema de programação inteira (PI), ou de programação inteira mista (PIM) (ambos de minimização), mede percentualmente a diferença entre o valor do óptimo inteiro do problema e o valor de um limite inferior desse problema. Esse limite inferior tanto pode ser representado pelo valor da solução óptima da relaxação linear, como pelo valor da solução óptima num qualquer nodo da árvore de pesquisa. O intervalo de integralidade é definido por: $\frac{Z_{PI} - Z_{LI}}{Z_{PI}}$, onde Z_{PI} é o valor da solução óptima inteira do problema de PI ou de PIM e Z_{LI} o valor de um limite inferior desse problema.

Em termos das principais vantagens da decomposição de Dantzig-Wolfe, destacam-se a obtenção de formulações mais fortes, que se traduzem em melhores "gaps" de integralidade e a potencial redução dos tempos de resolução dos problemas. Esta segunda vantagem assume maior importância quando se está perante um problema de grande dimensão.

3.2 Decomposição de Dantzig-Wolfe

Nesta secção apresenta-se a aplicação da decomposição de Dantzig-Wolfe a um problema de programação linear, constituído por um conjunto de restrições de ligação e um conjunto que tem uma estrutura especial.

Considere-se a seguinte formulação compacta de um problema de programação linear, ao qual chamamos de problema original, aqui designado por P :

$$Z_p = Min\, cx \qquad (P)$$

sujeito a:

$$Ax \geq b \qquad (3.1)$$

$$Bx \geq d \qquad (3.2)$$

$$x \geq 0\,,$$

onde c representa um vector linha de dimensão n, b um vector coluna de dimensão m, A uma matriz de dimensão $m \times n$, d um vector coluna de dimensão l, e por último B uma matriz de dimensão $l \times n$. Refere-se ainda que cada elemento de x está associado a uma variável de decisão denotada pelo índice j, x_j ($j=1,...,n$).

O conjunto de restrições *(3.1)* define as restrições de ligação e o conjunto de restrições *(3.2)* o conjunto que tem uma estrutura especial.

Seja o conjunto $S = \{\ x \in R^n : Bx \geq d,\ x \geq 0\ \}$ e $S \neq \varnothing$. De acordo com o teorema de Minkowski (ver por exemplo (Nemhauser and Wolsey 1999)) o poliedro S pode ser definido a partir de um número finito de pontos extremos e um número finito de raios extremos. Considerando que o poliedro S é constituído por H pontos extremos designados por P_k $(k=1,...,K)$ e por T raios extremos designados por P_r $(r=1,...,R)$, cada $x \in S$ pode ser representado como uma combinação convexa dos pontos extremos de S, mais uma combinação não negativa dos raios extremos de S :

$$x = \sum_{k=1}^{K} \lambda_k p_k + \sum_{r=1}^{R} \mu_r p_r$$

$$\sum_{k=1}^{K} \lambda_k = 1$$

$$\lambda_k \geq 0, \quad \forall k \in H$$

$$\mu_r \geq 0, \quad \forall r \in T\,.$$

No âmbito desta dissertação, interessa apenas analisar a aplicação da DDW ao caso em

que o conjunto *S* é um poliedro limitado. Assim, qualquer ponto $x \in S$ passa a ser representado como uma combinação convexa do número finito de pontos extremos de *S* :

$$x = \sum_{k=1}^{K} \lambda_k p_k \qquad (3.3)$$

$$\sum_{k=1}^{K} \lambda_k = 1$$

$$\lambda_k \geq 0 \quad \forall k \in H .$$

O problema *P* pode ser reformulado e passar a ser representado em termos de pontos extremos, através da substituição de *x* na sua formulação. Procedendo à substituição referida, obtém-se a seguinte formulação, equivalente à do problema original, aqui designada por *PM* :

$$Z_{PM} = Min \sum_{k=1}^{K} (cp_k)\lambda_k \qquad (PM)$$

sujeito a:

$$\sum_{k=1}^{K} (Ap_k)\lambda_k \geq b \qquad (3.4)$$

$$\sum_{k=1}^{K} \lambda_k = 1 \qquad (3.5)$$

$$\lambda_k \geq 0, \quad \forall k \in H .$$

As variáveis de decisão do problema mestre são as variáveis λ_k *(k=1,...,K)* e representam o peso de cada ponto extremo p_k *(k=1,...,K).* As restrições *(3.4)* definem as restrições de ligação, agora representadas em termos de pontos extremos e a restrição *(3.5)* define a restrição de convexidade, que garante a escolha de uma combinação convexa dos pontos extremos.

Qualquer solução válida para o problema *PM* é também válida para o problema original *P* e ambas as formulações têm um valor idêntico em termos de função objectivo na solução óptima. Através de uma solução válida qualquer ou da solução óptima do problema *PM*, pode fazer-se a leitura da solução correspondente no problema *P,* recorrendo à expressão *(3.3).*

Estabelecendo o paralelo entre o problema *PM* e o problema *P*, geralmente verifica-se que o primeiro tem um número extraordinariamente elevado de variáveis (igual ao número de pontos extremos do conjunto *H*), quando comparado com o segundo, mas substancialmente menos restrições que o segundo. De notar que o conjunto de restrições definidas pelo conjunto *S*, foi substituído por apenas uma restrição: a restrição de convexidade.

Dada a existência de um número exponencialmente grande de pontos extremos, a sua

enumeração explícita torna-se uma tarefa extremamente difícil ou mesmo impraticável. A solução óptima do problema mestre e consequentemente do problema original pode ser determinada sem ter de se enumerar explicitamente todas as variáveis. Na prática, existem muitos pontos extremos, associados a variáveis que têm valor nulo numa solução óptima do problema mestre, não sendo necessário incluí-las no problema *PM* para a obtenção da sua solução óptima.

Para lidar com o número exponencialmente grande de variáveis, pode então recorrer-se ao método de geração de colunas para resolver o problema mestre. Uma explicação mais detalhada a justificar o recurso a este método é apresentada na secção seguinte desta dissertação.

3.3 Método de Geração de Colunas

A geração de colunas é um método que pode ser aplicado eficientemente na resolução de problemas de PL e de PI de grande dimensão. No âmbito deste trabalho foca-se a sua aplicação à resolução do problema mestre *PM* definido na secção anterior.

Tal como já foi referido anteriormente, no problema mestre *PM*, existem muitas mais variáveis que na formulação original do problema *P*, uma vez que se associa uma nova variável a cada ponto extremo do conjunto *S*. O problema mestre fica assim definido por um número gigantesco de pontos extremos. Embora nos dias de hoje o armazenamento da informação não seja um problema devido aos excelentes progressos em termos de capacidade de memória dos computadores, torna-se de todo inconcebível armazenar os coeficientes de todos os pontos extremos de um problema de grande dimensão, com vista à obtenção da sua solução óptima. A principal motivação para o recurso à geração de colunas, prende-se então com a tentativa de resolução do problema mestre, sem ter de se enumerar explicitamente todos os pontos extremos a ele associados.

A ideia base subjacente à geração de colunas pode definir-se do seguinte modo: em vez de considerarmos todos os pontos extremos do conjunto *S* no problema mestre *PM*, vamos considerar apenas um conjunto restrito desses pontos extremos, definindo assim um problema mestre restrito PMR e avaliar se existem pontos extremos que não estão actualmente presentes no problema *PM*, que caso fossem incluídos no problema poderiam melhorar o valor da sua função objectivo. Os pontos que se revelarem atractivos devem então ser adicionados ao problema mestre.

Na geração de colunas, o PMR é optimizado iterativamente, até que exista a garantia de que a solução óptima do PMR é também a solução óptima do PM.

Em cada iteração do método, faz-se a optimização do PMR, obtendo-se uma solução que é óptima em termos das variáveis que nele estão incluídas. Contudo poderão existir variáveis, ou pontos extremos do conjunto *S*, que não façam parte actualmente do PMR que sejam atractivos. Para avaliar a atractividade destes pontos extremos temos de recorrer à solução óptima dual do PMR e à resolução de um subproblema.

Seja π o vector de variáveis duais associado ao conjunto de restrições *(3.4)* e θ a variável dual associada à restrição *(3.5)*.

O custo reduzido de uma variável λ_k é dado por: cp_k-πAp_k-θ. Se cp_k-πAp_k-$\theta \geq 0$, para $\forall k \in H$, a solução óptima do PMR, é também a solução óptima do problema mestre. Se, do conjunto de todas as variáveis que não pertencem ao PMR, existir pelo menos uma com custo reduzido negativo, então esta coluna deve ser adicionada ao PMR, e todo o processo deve ser repetido.

Como a solução óptima de um problema de PL, caso exista, corresponde a um ponto extremo, para se encontrar a variável λ_k que tem o menor custo reduzido, pode resolver-se o problema de PL seguinte:

$$Min\ Z_{SP} = (c - A\pi)x - \theta \qquad (SP)$$

sujeito a:

$$Bx \geq d$$

$$x \geq 0\ .$$

Este problema é habitualmente designado por subproblema (SP).

Assim, se o valor de $Z_{sp} \geq 0$, a solução óptima do PM foi encontrada. Se o valor de $Z_{sp} < 0$, uma nova coluna deve ser adicionada ao PMR e o procedimento apresentado, deverá ser repetido, até que não existam mais colunas atractivas.

Na Figura 9 sumariam-se os passos do método de geração de colunas.

Passo 1:	Definir um PMR válido, eventualmente com o recurso a variáveis artificiais e a alguns pontos extremos do conjunto S.
Passo 2:	Optimizar o PMR obtendo a sua solução óptima dual (π, θ).
Passo 3:	Resolver o subproblema SP e avaliar se existem pontos extremos do conjunto de pontos que não pertencem ao PMR que sejam atractivos: Se $Z_{sp} = (c-A\pi)x-\theta \geq 0 \Rightarrow$ Parar. A solução actual do PMR corresponde à solução óptima do PM; Caso contrário $\Rightarrow$ avançar para o Passo 4.
Passo 4:	Inserir uma nova coluna no PMR, correspondente à solução óptima do subproblema e voltar ao passo 2. Coluna a adicionar: $\begin{bmatrix} Ap_k \\ 1 \end{bmatrix}$

Figura 9: Algoritmo de geração de colunas.

3.4 Estrutura Angular em Blocos

Na secção 3.2 foi apresentada a aplicação da decomposição de Dantzig-Wolfe a um problema de programação linear, com um conjunto de restrições de ligação e um conjunto de restrições, que possui uma estrutura especial. Através das restrições de ligação foi definido um problema mestre e com base no segundo conjunto, foi definido um subproblema. Na secção 3.3 foi abordado um método de resolução eficiente desse problema mestre. Resta abordar o caso em que o segundo conjunto de restrições é constituído por vários subconjuntos de restrições, que correspondem a vários subsistemas. Nesta situação diz-se que o problema original tem uma estrutura angular em blocos (EAB), podendo ser representado da seguinte forma:

$$Z_{EAB} = Min\, c^1x^1 + c^2x^2 + \ldots + c^mx^m \qquad (EAB)$$

sujeito a:

$$A^1x^1 + A^2x^2 + \ldots + A^mx^m \geq b \qquad (3.6)$$

$$B^1x^1 \geq d^1 \qquad (3.7)$$

$$B^2x^2 \geq d^2 \qquad (3.8)$$

$$\ldots$$

$$B^mx^m \geq d^m \qquad (3.9)$$

$$x^1, \quad x^2, \quad \ldots \quad x^m \geq 0,$$

onde as restrições *(3.6)* são as restrições de ligação e as restrições *(3.7)*, *(3.8)* e *(3.9)* definem os

M conjuntos que têm uma estrutura especial. A letra *M* representa o conjunto de subsistemas, indexados por *j=1,..., m* e todos os subsistemas têm associado o mesmo tipo de problema.

Informações mais detalhadas acerca da estrutura angular em blocos podem ser encontradas na secção 3.1. Nesta secção, apresenta-se a aplicação da decomposição de Dantzig-Wolfe ao problema EAB.

Relativamente ao caso já abordado, as principais diferenças no caso de termos um problema com uma estrutura angular em blocos são o facto de agora passarmos a ter vários subproblemas (*M* subproblemas para o caso do problema EAB), um por cada subsistema existente e o facto de, no problema mestre, passar a figurar uma restrição de convexidade por cada subproblema.

Assumindo que $S^j = \{x^j \in IR^n : B^j x^j \geq d^j,\ x^j \geq 0\}$ para *j=1,..., m* são conjuntos limitados, qualquer ponto $x^j \in S^j$ pode ser expresso como uma combinação convexa dos H^j pontos extremos de S^j:

$$x^j = \sum_{k=1}^{K} \lambda_k^j p_k^j, \quad \forall j \in M \qquad (3.10)$$

$$\sum_{k=1}^{K} \lambda_k^j = 1, \quad \forall j \in M$$

$$\lambda_k^j \geq 0, \quad \forall k \in H^j, \forall j \in M .$$

Substituindo x^j no problema *EAB*, obtém-se o problema mestre *PMB*:

$$Z_{PMB} = Min \sum_{j=1}^{m} \sum_{k=1}^{K} \left(c^j p_k^j\right) \lambda_k^j \qquad (PMB)$$

sujeito a:

$$\sum_{j=1}^{m} \sum_{k=1}^{K} \left(A^j p_k^j\right) \lambda_k^j \geq b \qquad (3.11)$$

$$\sum_{k=1}^{K} \lambda_k^j = 1, \quad \forall j \in M \qquad (3.12)$$

$$\lambda_k^j \geq 0, \quad \forall k \in H^j, \forall j \in M .$$

As variáveis de decisão do problema mestre são as variáveis λ_k^j ($\forall k \in H^j, \forall j \in M$) e representam o peso do ponto extremo p_k *(k=1,...,K)* do subproblema *j*. As restrições *(3.11)* são as restrições de ligação, agora representadas em termos de pontos extremos e as restrições *(3.12)* são as *M* restrições de convexidade.

Quando o problema original tem uma estrutura angular em blocos, passam a existir e a ter de se resolver *M* subproblemas.

Se π representar o vector de variáveis duais associado ao conjunto de restrições *(3.11)*, e θ^j a variável dual associada à restrição *j* do conjunto de restrições de convexidade *(3.12)*, os *M* subproblemas a resolver (que são idênticos ao subproblema *SP* definido na secção 3.3) são:

$$Z_{SPB^j} = Min\left(c^j - A^j\pi\right)x^j - \theta^j \qquad (SPB^j)$$

sujeito a:

$$B^j x^j \geq d^j$$

$$x^j \geq 0\,.$$

A solução óptima de cada subproblema permitirá avaliar a atractividade das colunas que não estão presentes no *PMR*, podendo-se inserir no PMR uma coluna por cada subproblema que apresente um custo reduzido (representado pelo valor de $Z_{SPB}{}^j$) negativo.

Para resolver o problema mestre *PMB* recorre-se habitualmente ao método de geração de colunas, já definido na secção 3.3. Na Figura 10, apresenta-se uma reformulação do algoritmo a aplicar no método de geração de colunas, definido na Figura 9, para o caso em que o problema original apresenta uma estrutura angular em blocos.

Passo 1:	Definir um PMR válido, eventualmente com o recurso a variáveis artificiais e a alguns pontos extremos dos conjuntos S^j.
Passo 2:	Optimizar o PMR obtendo a sua solução óptima (π, θ^j).
Passo 3:	Resolver os *M* subproblemas SPB^j e avaliar se existem pontos extremos do conjunto de pontos que não pertencem ao PMR que sejam atractivos: Se para os M subproblemas: $Z_{spb}{}^j = (c^j\text{-}A^j\pi)x^j\text{-}\theta^j \geq 0 \Rightarrow$ Parar. A solução actual do PMR corresponde à solução óptima do PMB; Caso contrário $\Rightarrow$ avançar para o Passo 4.
Passo 4:	Para cada subproblema com $Z_{spb}{}^j = (c^j\text{-}A^j\pi)x^j\text{-}\theta^j < 0 \Rightarrow$ Inserir uma nova coluna no PMR, correspondente à solução óptima do subproblema *j* e voltar ao passo 2.

Figura 10: Algoritmo de geração de colunas - problema com EAB.

3.5 Método de Partição e Geração de Colunas

Vimos, na secção anterior, como aplicar a decomposição de Dantzig-Wolfe a um problema de programação linear com uma estrutura angular em blocos, problema *EAB*, e como resolver esse problema através do método de geração de colunas. Consideremos agora o problema EABI, que resulta da adição das condições de integralidade ao problema EAB, que foi já apresentado na secção 3.4. Neste caso, ao aplicarmos a decomposição de Dantzig-Wolfe e ao resolvermos o problema reformulado (problema mestre) através do método de geração de colunas, não existe a garantia de que na solução óptima todas as variáveis que devem obedecer às condições de integralidade, sejam efectivamente inteiras. Poderá ser então necessário recorrer a um método de enumeração, para encontrar a solução óptima inteira do problema EABI. O valor da solução que se obtém ao resolver a relaxação do problema mestre é um limite inferior (num problema de minimização) para o valor do óptimo inteiro.

Se aplicarmos o método de partição e avaliação ao problema mestre restrito e considerarmos apenas os pontos extremos incluídos nesse PMR, não existe a garantia de que a solução óptima inteira do problema original seja encontrada. Pode acontecer que num determinado nodo da árvore de pesquisa, após se ter introduzido a restrição de partição desse nodo, exista um ponto extremo que seja atractivo, mas que não está presente no PMR actual. Para que se encontre a solução óptima inteira desse nodo, será necessário gerar esse ponto extremo, sem o qual a solução óptima inteira não será encontrada. Para resolver o problema inteiro, utiliza-se então um procedimento baseado na combinação do método de geração de colunas com o método de partição e avaliação. A combinação destes dois métodos, dá origem a uma técnica conhecida por método de partição e geração de colunas, ou por "branch and price", na literatura anglo-saxónica. Em (Barnhart, Johnson et al. 1998) apresentam-se em detalhe os conceitos fundamentais desta técnica.

A filosofia do método de partição e geração de colunas assenta na resolução de um problema mestre em cada nodo da árvore de pesquisa através do método de geração de colunas.

Se a solução óptima de um determinado nodo da árvore não for inteira e se o valor da função objectivo nesse nodo não for maior que o valor da solução incumbente actual, deverão adicionar-se restrições de partição a esse nodo. A adição das restrições de partição visa eliminar a solução fraccionária actual através da divisão do espaço de soluções válidas.

Existem várias alternativas que se podem adoptar na definição das restrições de partição. Uma regra de partição possível é a que se baseia nas variáveis de peso, λ_k^j. Nesta regra de

partição, após se ter determinado a solução óptima do problema mestre na raiz da árvore de pesquisa, se todas as variáveis λ^j_k tiverem valores inteiros, a solução óptima inteira do correspondente problema *EABI* foi encontrada. Caso exista uma ou mais variáveis de peso com valores fraccionários, por forma a determinar a solução óptima inteira do problema EABI, num esquema de partição simples, deverão adicionar-se as seguintes restrições de partição ao problema mestre:

$$\lambda^j{}_{\bar{k}} \leq \left\lfloor \bar{\lambda}^j{}_{\bar{k}} \right\rfloor \; e \; \lambda^j{}_{\bar{k}} \geq \left\lfloor \bar{\lambda}^j{}_{\bar{k}} \right\rfloor + 1,$$

onde $\bar{\lambda}^j{}_{\bar{k}}$ representa o valor actual da variável fraccionária e $\lambda^j{}_{\bar{k}}$ a variável que é fraccionária.

Estas duas novas restrições, forçam a criação de dois novos problemas ou dois novos nodos (constituídos pelo PMR actual e por uma das restrições de partição acima apresentadas), que correspondem ao primeiro nível da árvore de pesquisa. Estes problemas devem ser resolvidos até à optimalidade, e caso ainda existam variáveis de peso fraccionárias na sua solução óptima, o procedimento descrito no parágrafo anterior deverá ser repetido, até que exista a garantia de que a solução óptima inteira do problema EABI foi encontrada.

Esta estratégia de partição apresenta alguns inconvenientes, que tornam a sua utilização pouco apropriada. Uma das suas principais desvantagens relaciona-se com o facto de, após a introdução das restrições de partição no problema mestre, poder ser necessário fazer alterações nos subproblemas, alterações essas que podem levar a que a sua estrutura se altere. Em termos práticos, podemos passar de um subproblema fácil para um subproblema sem estrutura bastante mais difícil de resolver. No processo de geração de colunas, como os subproblemas são resolvidos um elevado número de vezes, é de todo importante para a eficiência do algoritmo que estes subproblemas não tenham uma grande exigência em termos de carga computacional, sendo fundamental garantir que a estrutura dos subproblemas não se altera quando se passa da raiz da árvore de pesquisa para os nodos subsequentes da árvore, que integram já restrições de partição. Uma outra desvantagem desta estratégia de partição tem a ver com o facto de poder conduzir a árvores de pesquisa desequilibradas. Em (Barnhart, Johnson et al. 1998) e (Vanderbeck 1994) esta e outras estratégias de partição são analisadas em detalhe.

Nesta dissertação, aborda-se uma regra de partição baseada nas variáveis originais.

Quer na raiz da árvore de pesquisa, quer em qualquer nodo da árvore de pesquisa, é possível a partir de uma solução óptima do problema mestre, fazer-se a leitura da solução correspondente em termos das variáveis de decisão originais do problema EABI, variáveis x^j,

recorrendo à expressão *(3.10)* apresentada na página 35. Caso todas essas variáveis sejam inteiras, a solução óptima inteira do problema EABI foi encontrada. Contudo, se existir uma ou mais variáveis x^j que sejam fraccionárias, deverão adicionar-se ao problema duas restrições de partição. Estas duas restrições devem forçar a variável que é actualmente fraccionária, ou uma das variáveis que é actualmente fraccionária, a tomar valores inteiros. Na regra de partição baseada nas variáveis originais, as restrições que se devem adicionar ao problema, da raiz da árvore ou de qualquer nodo da árvore cuja solução óptima seja fraccionária e o seu valor correspondente não seja maior que o valor da solução incumbente actual, são:

$$\sum_{k\in\overline{H}} \overline{\lambda}_k^j p_k^j \leq \lfloor \overline{x}^j \rfloor \; e \; \sum_{k\in\overline{H}} \overline{\lambda}_k^j p_k^j \geq \lfloor \overline{x}^j \rfloor + 1,$$

onde $\overline{H}$ representa o conjunto de pontos extremos que fazem parte do problema mestre restrito actual e $\overline{x}^j$ representa o valor actual da variável original que é fraccionária.

A introdução destas duas restrições cria dois novos nodos na árvore de pesquisa, que devem ser optimizados e caso a sua solução óptima seja fraccionária, todo o processo anteriormente descrito deve ser repetido, até que a melhor solução inteira seja encontrada.

Esta regra de partição tem a vantagem de ser geral, podendo ser aplicada a qualquer tipo de problema. Fazendo a partição nas variáveis originais no problema mestre, as dificuldades que advêm da aplicação da regra de partição baseada nas variáveis de peso deixam de existir, existindo a garantia de que a estrutura dos subproblemas não se altera. Contudo, face à introdução das restrições de partição no problema mestre de um determinado nodo da árvore de pesquisa, para que o custo reduzido dos subproblemas seja correctamente determinado, as variáveis duais associadas às restrições de partição desse nodo, devem passar a integrar a função objectivo dos subproblemas.

Conceptualmente, quando se utiliza o método de partição e geração de colunas, em cada nodo da árvore de pesquisa, aplica-se a decomposição de Dantzig-Wolfe ao problema EABI e adicionam-se restrições de partição no problema mestre que resulta da decomposição, sendo depois esse problema mestre resolvido por geração de colunas.

Nesta secção considerou-se a aplicação do método de partição e geração de colunas a um problema de programação inteira pura. A extensão para o caso de um problema de programação inteira mista (que pode ser constituído por simultaneamente variáveis contínuas e/ou inteiras e/ou binárias) é imediata e será apresentada nas subsecções 4.2.4 e 4.3.4, aplicada ao problema LPMACC.

4 Decomposições para o Problema LPMACC

Neste capítulo apresentam-se duas decomposições de Dantzig-Wolfe para o problema LPMACC, ambas baseadas na formulação clássica do problema, já apresentada na subsecção 2.3.2.

Na secção 4.1 faz-se uma introdução ao capítulo, nomeadamente às duas decomposições referidas no parágrafo anterior, designadas por decomposição por artigo e por decomposição por período.

Na secção 4.2 aplica-se a decomposição por artigo ao problema LPMACC. Nesta decomposição as restrições de capacidade da formulação clássica representam as restrições de ligação. Nas subsecções 4.2.1, 4.2.2 e 4.2.3 define-se o problema mestre, os subproblemas e o primeiro problema mestre restrito que se obtêm aplicando a decomposição por artigo, e, na subsecção 4.2.4, definem-se as regras de partição para a obtenção da solução óptima inteira do problema.

Na secção 4.3 aplica-se a decomposição por período também ao LPMACC, mas neste caso são as restrições de equilíbrio de stocks da formulação clássica que definem as restrições de ligação. Nas subsecções 4.3.1, 4.3.2 e 4.3.3 define-se o problema mestre, os subproblemas e o primeiro problema mestre restrito que se obtêm aplicando a decomposição por período, e, na subsecção 4.3.4, definem-se as regras de partição para a obtenção da solução óptima inteira do problema, quando se aplica esta decomposição.

Para exemplificar a aplicação das duas decomposições, recorre-se nas subsecções 4.2.4 e 4.3.4 a uma pequena instância do problema LPMACC com 2 produtos e 3 períodos, onde se apresenta a resolução do problema mestre e do problema inteiro dessa instância. Para resolver o problema mestre, recorre-se ao algoritmo de geração de colunas, já apresentado, e, para resolver o problema inteiro, recorre-se ao método de partição e geração de colunas.

4.1 Introdução

A estrutura da formulação clássica do problema LPMACC, enquadra-se na estrutura angular em blocos definida na secção 3.4 desta dissertação. Por forma a tirar partido da estrutura do problema e com o intuito da pesquisa de formulações e métodos de resolução eficientes para o problema, apresentam-se neste capítulo duas decomposições de Dantzig-Wolfe, aplicadas à formulação clássica do LPMACC. Ambas as decomposições permitem a obtenção de formulações mais fortes, com limites inferiores de melhor qualidade que os produzidos pela resolução da formulação clássica.

Na primeira decomposição, à qual chamamos decomposição por artigo, os subproblemas são definidos através das restrições de equilíbrio de stocks e das restrições que garantem a ocorrência de uma preparação sempre que se produza, obtendo-se deste modo um subproblema para cada artigo. Esta decomposição foi já estudada por vários autores. Na secção 2.4, apresentaram-se as principais contribuições e os métodos de resolução por eles utilizados. Nesta dissertação, apresenta-se para esta decomposição uma nova regra de partição para a obtenção da solução óptima inteira do problema.

Na segunda decomposição, que designamos por decomposição por período, os subproblemas são definidos através das restrições de capacidade e das restrições que garantem a ocorrência de uma preparação sempre que se produza. Nesta decomposição, o número de subproblemas passa a estar associado ao número de períodos de planeamento, existindo um subproblema para cada período. Quanto às restrições de equilíbrio de stocks, nesta decomposição, estas definem as restrições de ligação e ficam no problema mestre. (Diaby, Bahl et al. 1992a) abordaram uma relaxação de lagrange, baseada na relaxação das restrições de equilíbrio de stocks. Para resolver o problema reformulado, os autores utilizaram o método do subgradiente. O trabalho que aqui se desenvolve difere do anterior, na medida em que os métodos que se utilizam na resolução do problema são diferentes. Para resolver o problema LPMACC, aplica-se a DDW e resolve-se o problema reformulado através do método de geração de colunas. Para encontrar a solução óptima inteira do problema, utiliza-se uma regra de partição baseada nas variáveis originais e aplica-se o método de partição e geração de colunas.

Em termos teóricos, sabe-se que aplicando a DDW a um problema de minimização, o limite inferior que se obtém ao resolver a sua relaxação linear pode ser de melhor qualidade que o obtido pela relaxação linear da formulação original, se o subproblema não tiver a propriedade da integralidade (Geoffrion 1974). Quando o subproblema tem a propriedade da integralidade, ou seja quando todos os pontos extremos do poliedro definido pelo subproblema são inteiros, o

limite inferior produzido pela DDW tem um valor igual ao produzido pela solução óptima da relaxação linear do problema original.

Ao resolver o problema mestre que resulta da decomposição por artigo ou da decomposição por período, através do método de geração de colunas, não existe a garantia de que a solução óptima inteira do problema seja encontrada, pelo que será necessário recorrer a um método de enumeração, para determinar essa solução óptima. Em ambas as decomposições, para resolver o problema inteiro utiliza-se o método de partição e geração de colunas.

4.2 Decomposição por Artigo

Na formulação clássica do problema LPMACC, apenas as restrições de capacidade interligam os artigos a produzir. Se se deixar de considerar estas restrições, o problema decompõe-se num conjunto de (sub)problemas independentes, cada um correspondendo a um problema de lotes de produção de artigo único com capacidade ilimitada, conhecido na literatura anglo-saxónica por "Uncapacitated Lot-sizing Problem".

Para tirar partido da estrutura apresentada no parágrafo anterior, e estabelecendo o paralelo com a estrutura angular em blocos apresentada na secção 3.4, na decomposição por artigo, as restrições de capacidade da formulação clássica *(2.3)* (página 16) definem as restrições de ligação, que ficam no problema mestre, e as restrições de equilíbrio de stocks *(2.1)* e *(2.2)*, as restrições que relacionam as variáveis de preparação com as variáveis de produção *(2.4)* e as restrições *(2.5)*, *(2.6)*, *(2.7)* e *(2.8)* definem os subproblemas. Aplicando a DDW à formulação clássica do problema, na decomposição por artigo, obtêm-se I subproblemas de lotes de produção de artigo único com capacidade ilimitada, um para cada artigo.

Nesta decomposição, os pontos extremos dos subproblemas correspondem a planos de produção. Para um determinado artigo, cada plano de produção indica em que períodos se deve produzir e em que quantidades.

4.2.1 Definição do Problema Mestre

Para a definição do problema mestre, considerem-se os parâmetros e variáveis definidos na secção 2.3.2, relativos à formulação clássica, e o seguinte conjunto, parâmetros e variável de decisão:

Conjunto:

H representa o conjunto de planos de produção indexados por *k=1,...,t*

Parâmetros:

β^{i}_{jk} assume o valor *1* caso exista uma preparação do artigo *i* no período *j* no plano de produção *k*, e assume o valor *0* caso contrário; *i=1,...,n ; j=1,...,m ; k=1,...t*

α^{i}_{jk} representa a quantidade a produzir do artigo *i* no período *j*, no plano de produção *k*; *i=1,...,n ; j=1,...,m ; k=1,...t*

δ^{i}_{jk} representa o nível de inventário do artigo *i* no final do período *j*, no plano de produção *k*; *i=1,...,n ; j=1,...,m ; k=1,...t*

Variável de Decisão:

z^{i}_{k} representa o peso do plano de produção *k* do artigo *i; i=1,...,n ; k=1,...t*

A solução da relaxação linear do problema LPMACC, quando se aplica a decomposição por artigo, pode ser obtida resolvendo o seguinte problema mestre, aqui designado por *PMI*:

$$Z_{PMI} = Min \sum_{i=1}^{n} \sum_{k=1}^{t} \left[\left(\sum_{j=1}^{m} q^{i}_{j} \beta^{i}_{jk} + p^{i}_{j} \alpha^{i}_{jk} + h^{i}_{j} \delta^{i}_{jk} \right) z^{i}_{k} \right] \qquad (PMI)$$

Sujeito a:

$$\sum_{k=1}^{t} z^{i}_{k} = 1, \quad \forall i \in I \qquad (\pi^{i}) \qquad (4.1)$$

$$\sum_{i=1}^{n} \sum_{k=1}^{t} (b^{i}_{j} \beta^{i}_{jk} + a^{i}_{j} \alpha^{i}_{jk}) z^{i}_{k} \le c_{j}, \quad \forall j \in J \quad (\mu_{j}) \qquad (4.2)$$

$$z^{i}_{k} \ge 0, \quad \forall i \in I, \forall k \in H. \qquad (4.3)$$

As variáveis de decisão z^{i}_{k} representam a proporção da procura do artigo *i* que é satisfeita através do plano de produção *k*. A função objectivo minimiza a soma dos custos totais. As restrições *(4.1)* são as restrições de convexidade. Existe uma restrição de convexidade para cada artigo, na qual se força a escolha de uma combinação de planos de produção. Associada a cada restrição de convexidade está uma variável dual π^{i}. O segundo conjunto de restrições, conjunto *(4.2)*, diz respeito às restrições de capacidade. Neste conjunto força-se a que a combinação de planos de produção escolhida respeite a capacidade disponível em cada período. Este segundo

conjunto tem associadas as variáveis duais μ_j. As restrições *(4.3)* forçam as variáveis de decisão a tomarem valores não negativos.

Em termos de dimensão, o problema *PMI*, comparativamente com o problema *C*, tem um número mais reduzido de restrições. Enquanto que na formulação clássica existem $J+2\times I\times J$ restrições, no problema mestre *PMI*, este número reduz-se para $I+J$, onde I representa o número de artigos e J o número de períodos de planeamento. Já em termos de variáveis, o problema *PMI*, comparativamente com o problema *C*, tem um número gigantesco de variáveis, pelo que a utilização do método de geração de colunas na resolução do problema mestre *PMI* é de todo conveniente.

Caso o problema mestre não fosse resolvido por geração de colunas, este seria constituído por $I\times 2^{J-1}$ variáveis, uma vez que cada artigo tem associados 2^{J-1} planos de produção. Para um problema com por exemplo, 30 artigos diferentes e 20 períodos de planeamento, teríamos de resolver um problema com cerca de 16 milhões de variáveis. Seria incomportável lidar ou resolver eficientemente um problema no qual o número de variáveis cresce exponencialmente com o número de períodos, sem recorrer a um método de enumeração implícita, por exemplo ao método de geração de colunas.

A partir de uma solução do problema *PMI* é possível recuperar a solução correspondente do problema clássico *C*, através das expressões:

$$x_j^i = \sum_{k=1}^{t} \alpha_{jk}^i z_k^i \qquad (4.4)$$

$$y_j^i = \sum_{k=1}^{t} \beta_{jk}^i z_k^i \qquad (4.5)$$

Conhecido o valor das variáveis x_j^i, através das equações *(2.1)* e *(2.2)*, é imediata a determinação do valor das variáveis de decisão auxiliares s_j^i.

Até agora, foi apresentado o problema mestre que resulta da aplicação da decomposição de Dantzig-Wolfe à formulação clássica, quando se decompoem as restrições de capacidade. Na subsecção seguinte, definem-se os subproblemas que resultam desta decomposição e na subsecção 4.2.3, apresenta-se a metodologia a adoptar para a resolução do problema mestre.

4.2.2 Subproblemas

Na decomposição por artigo, cada subproblema está associado ao problema de lotes de produção de artigo único com capacidade ilimitada e existem I subproblemas a resolver.

Se μ_j representar o vector de variáveis duais associado ao conjunto de restrições de capacidade *(4.2)* e π^i a variável dual associada à restrição *i* do conjunto de restrições de convexidade *(4.1)*, os *I* subproblemas a resolver são definidos por:

$$Z_{SPI} = Min \sum_{j=1}^{m} p_j^i x_j^i + \sum_{j=1}^{m} q_j^i y_j^i + \sum_{j=1}^{m} h_j^i s_j^i - \sum_{j=1}^{m} (b_j^i y_j^i)\mu_j - \sum_{j=1}^{m} (a_j^i x_j^i)\mu_j - \pi^i$$

(SPI)

Sujeito a:

$$x_1^i = d_1^i + s_1^i \qquad (4.6)$$

$$s_{j-1}^i + x_j^i = d_j^i + s_j^i, \quad \forall j \in J \setminus \{1\} \qquad (4.7)$$

$$x_j^i \leq min\left\{\frac{c_j - b_j^i}{a_j^i}, \sum_{t=j}^{m} d_t^i\right\} y_j^i, \quad \forall j \in J \qquad (4.8)$$

$$x_j^i \geq 0, \quad \forall j \in J \qquad (4.9)$$

$$y_j^i \in \{0,1\}, \quad \forall j \in J \qquad (4.10)$$

$$s_j^i \geq 0, \quad \forall j \in J \setminus \{m\} \qquad (4.11)$$

$$s_m^i = 0\,. \qquad (4.12)$$

Na função objectivo procura-se o plano de produção que minimiza o custo reduzido. Calcular o custo reduzido equivale a resolver um problema de lotes de produção multi-artigo capacitado, no qual as restrições de capacidade foram substituídas pelos seus preços marginais. As restrições de equilíbrio de stocks *(4.6)* e *(4.7)*, as restrições que garantem que uma preparação ocorre se o artigo *i* for produzido no período *j (4.8)* e as restrições *(4.9)*, *(4.10)*, *(4.11)* e *(4.12)* definem o subproblema.

O subproblema *SPI* tem apenas pontos extremos (ou planos de produção). Cada subproblema tem associados 2^{J-1} planos de produção, pelo que a solução óptima de cada um dos *I* subproblemas representa um plano de produção relativo ao artigo *i*. Cada plano de produção, indica para um determinado artigo a quantidade a produzir desse artigo e em que período(s) deve ser produzida. Como no subproblema não existem restrições de capacidade, em todos os planos de produção, quando se toma a decisão de produzir num determinado período, produz-se para satisfazer a procura de um número inteiro de períodos. Estes planos são conhecidos por planos de Wagner-Whitin (Wagner and Whitin 1958) e obedecem à propriedade de Wagner-Whitin. De acordo com esta propriedade, para satisfazer a procura de um determinado artigo, nunca se produz esse artigo num determinado período *j* e ao mesmo tempo se utilizam unidades

desse artigo, provenientes do período imediatamente anterior ao período *j*. A mesma propriedade implica ainda que, quando se toma a decisão de produzir, se produza para um número inteiro e consecutivo de períodos.

Para determinar eficientemente a solução óptima destes *I* subproblemas, em cada iteração do método de geração de colunas, pode recorrer-se ao algoritmo de programação dinâmica proposto por Wagner e Whitin (Wagner and Whitin 1958). Seja x^{i*}_j, y^{i*}_j e s^{i*}_j a solução óptima do subproblema relativo ao artigo *i.* Se o custo reduzido desta solução for menor do que zero, ou seja se:

$$\sum_{j=1}^{m} q_j^i y_j^{i*} + \sum_{j=1}^{m} p_j^i x_j^{i*} + \sum_{j=1}^{m} h_j^i s_j^{i*} - \sum_{j=1}^{m} \left(b_j^i y_j^{i*}\right)\mu_j - \sum_{j=1}^{m} \left(a_j^i x_j^{i*}\right)\mu_j - \pi^i < 0,$$

existe uma coluna atractiva, ou plano de produção, que deverá ser inserido no problema mestre. Em cada iteração do método de geração de colunas, podem portanto ser adicionadas *I* novas colunas ao problema mestre, caso cada subproblema contribua com uma coluna atractiva.

4.2.3 Primeiro Problema Mestre Restrito

Para resolver o problema mestre *PMI* definido na subsecção 4.2.1, recorre-se ao método de geração de colunas, sendo este problema resolvido iterativamente até que exista a garantia de que a sua solução óptima foi encontrada. Para iniciar o processo iterativo, torna-se necessário definir um problema mestre, constituído por um número reduzido de variáveis, designado habitualmente por problema mestre restrito.

Este problema mestre restrito tem de ser um problema válido, para que se possa iniciar o processo de geração de colunas, visto ter de fornecer variáveis duais aos subproblemas, para fazer a avaliação dos planos de produção atractivos. Para tornar o primeiro problema mestre restrito da decomposição por artigo num problema válido, inclui-se nas restrições de convexidade desse PMR um conjunto de variáveis artificiais, v^i, associadas a cada artigo. Face à introdução das variáveis artificiais no problema mestre restrito, existem algumas alterações a fazer na sua formulação matemática. Na função objectivo do problema mestre definido na subsecção 4.2.1 deverá adicionar-se o seguinte termo: $\sum_{i=1}^{n} f^i v^i$ e a cada restrição de convexidade *(4.1)* deve adicionar-se uma variável artificial v^i. Cada restrição de convexidade do PMR passa portanto a ter associada uma variável artificial com um coeficiente elevado na função objectivo, que torna qualquer solução com variáveis artificiais positivas de pior qualidade que uma solução óptima em termos das variáveis z^i_k. Nesta estratégia, inclui-se ainda no primeiro PMR um subconjunto de planos de produção do conjunto de todos os planos de

produção dos *I* subproblemas.

Para além destas variáveis artificiais, que visam validar o problema, podem acrescentar-se um outro tipo de variáveis artificiais ao primeiro PMR, associadas às restrições de capacidade. Por cada restrição de capacidade é introduzida uma variável artificial v_j, sendo introduzidas no total *J* variáveis artificiais, uma por cada período. Estas variáveis podem ser vistas como um acréscimo na capacidade de cada período de planeamento e permitem seleccionar qualquer combinação de planos de produção, pois as restrições de capacidade deixam de ser efectivas. As alterações à formulação matemática do PM neste caso são as seguintes: na função objectivo do problema mestre definido na subsecção 4.2.1 deverá adicionar-se o termo $\sum_{j=1}^{m} f_j v_j$ e no termo independente de cada restrição de capacidade *(4.2)*, deve adicionar-se uma variável artificial v_j. Sempre que a variável v_j tomar um valor positivo, será necessário pagar um custo elevado f_j, garantindo-se deste modo que apenas se recorrerá ao acréscimo de capacidade, quando a capacidade disponível num determinado período não for suficiente para fazer face à procura.

4.2.4 Regras de Partição

Ao resolver o problema mestre definido na subsecção 4.2.1 está-se a resolver a raiz da árvore de pesquisa do método de partição e geração de colunas, ou ainda, a determinar a solução óptima da relaxação linear da decomposição por artigo. Esta solução apresenta um limite inferior para o valor do óptimo inteiro, que é melhor ou igual que o obtido pela solução da relaxação linear da formulação clássica, já que os subproblemas não têm a propriedade da integralidade (Geoffrion 1974).

A solução óptima que se obtém ao resolver a relaxação linear da decomposição por artigo poderá não corresponder à solução óptima inteira da formulação clássica, apresentada na secção 2.3.2. Ao passar da solução óptima em termos das variáveis z^i_k, para a solução em termos das variáveis x^i_j, y^i_j e s^i_j, poderão existir variáveis y^i_j que não sejam binárias, sendo, nesse caso, necessário forçar estas variáveis a tomar valores binários. É nessa perspectiva que se utiliza aqui o método de partição e geração de colunas, apresentado na secção 3.5. A estratégia de partição que se utiliza é a que se baseia nas variáveis originais, também apresentada na secção 3.5.

Com base na solução óptima do problema mestre da decomposição por artigo, tal como já referido, pode recuperar-se a solução correspondente da formulação clássica, em termos das variáveis originais, recorrendo às expressões *(4.4)* e *(4.5)*. Caso as variáveis de preparação, as variáveis y^i_j, não sejam binárias, isto é, se alguma restrição do conjunto de restrições de

integralidade (conjunto de restrições *(2.8)* da página 16) for violada, será necessário recorrer ao método de partição e geração de colunas para determinar a solução óptima inteira do problema LPMACC.

Quando a solução da raiz da árvore de pesquisa, ou a solução de um determinado nodo não é inteira, e existe interesse em explorá-lo, devem adicionar-se dois novos nodos ao problema. Cada um destes nodos passa a definir um "novo" problema, no qual para além de se considerar o problema mestre actual, se considera também uma das restrições de partição, a seguir apresentadas:

$$\sum_{k \in \overline{H}} \beta_{jk}^{i} z_k^i = 0$$

$$\sum_{k \in \overline{H}} \beta_{jk}^{i} z_k^i = 1$$

onde $\overline{H}$ representa o conjunto de planos de produção que fazem parte desse problema mestre actual.

A primeira restrição de partição força a variável de preparação cujo valor actual é fraccionário a tomar o valor *0* e a segunda restrição de partição força-a a tomar o valor *1*.

Face à introdução das restrições de partição, o espaço de soluções válidas divide-se e criam-se dois sub-domínios. Em cada um dos novos nodos, deve adoptar-se o seguinte procedimento:

- Aplicar o método de geração de colunas;
- Analisar a solução óptima em termos das variáveis originais;
- Se existirem nessa solução, variáveis de preparação com valor fraccionário adicionar as restrições de partição ao problema mestre e passar para um novo nodo da árvore de pesquisa, senão parar.

No procedimento a aplicar em cada nodo, descrito no parágrafo anterior, vimos que, após a introdução da restrição de partição o nodo em questão deve ser optimizado, recorrendo-se para o efeito ao método de geração de colunas. Como neste nodo existe uma nova restrição, para que os custos reduzidos dos subproblemas a resolver no processo de geração de colunas sejam correctamente determinados, torna-se necessário alterar a função objectivo desses subproblemas, na qual se deve passar a considerar a variável dual associada à restrição de

partição que foi adicionada. Se η^i_j representar a variável dual associada à restrição de partição que foi adicionada ao nodo, na função objectivo de cada um dos *I* subproblemas, apresentada na subsecção 4.2.2, deve subtrair-se o termo seguinte:

$$\sum_{j=1}^{m} y^i_j \eta^i_j ,$$

para que os custos reduzidos no subproblema sejam correctamente determinados.

Adoptando a estratégia de partição baseada nas variáveis originais, existe a garantia de que a estrutura do subproblema se mantém, sendo necessário apenas efectuar a alteração que acabámos de analisar, na função objectivo dos subproblemas.

Na decomposição por artigo, não é de todo conveniente a utilização da regra de partição baseada nas variáveis de peso, que foi apresentada na secção 3.5. Como vimos, cada plano de produção, nesta decomposição, determina para um dado artigo a quantidade a produzir desse artigo em cada período, estando por isso a variável z^i_k associada a um plano de produção. Ao adicionarmos uma restrição de partição ao PM, baseada na variável z^i_k, temos de adicionar à função objectivo do subproblema a variável dual desta restrição de partição. Como esta variável dual está associada a um plano de produção, torna-se complexo adicionar o seu valor à função objectivo do subproblema, porque as variáveis deste estão associadas a períodos e produtos, e não a planos de produção. Fazendo a partição baseada nas variáveis originais, esta dificuldade é ultrapassada.

4.2.5 Exemplo

Para ilustrar a decomposição por artigo, recorre-se novamente ao Exemplo apresentado na subsecção 2.3.4 desta dissertação, relativo a uma instância do problema LPMACC com 2 artigos e 3 períodos de planeamento. Nos parágrafos seguintes, aborda-se a resolução do problema reformulado, que corresponde a resolver a raiz da árvore de pesquisa e posteriormente exemplifica-se a utilização do método de partição e geração de colunas, para determinar a solução óptima inteira do Exemplo.

Na Figura 11 apresenta-se uma reestruturação da formulação clássica do Exemplo já apresentado na subsecção 2.3.4 na Figura 5. Nesta reestruturação as variáveis e restrições do problema ordenam-se de uma forma diferente, para tornar clara a existência de uma estrutura angular em blocos. Na Figura 11 apresenta-se ainda a solução óptima da relaxação linear do

Exemplo na última linha da figura. O custo total desta solução é de 1 161 unidades monetárias (u. m.).

Através da análise da figura, torna-se evidente a existência de 1 conjunto de restrições que relaciona os 2 artigos (bloco 3) e de outros 2 conjuntos, associados apenas a um dos artigos (bloco 1 e 2). A estrutura dos blocos 1 e 2 é similar, diferindo apenas no artigo a que diz respeito.

A partir desta formulação, define-se um problema mestre, onde figuram as restrições de ligação (bloco 3) e 2 subproblemas de lotes de produção de artigo único com capacidade ilimitada, associados a cada um dos artigos.

		v1	**s11**	**s12**	**s13**	**x11**	**x12**	**x13**	**y11**	**y12**	**y13**	**v2**	**s21**	**s22**	**s23**	**x21**	**x22**	**x23**	**y21**	**y22**	**y23**		
Bloco 1	i=1, j=1	1	-1			1																=	100
	i=1, j=2		1	-1			1															=	80
	i=1, j=3			1	-1			1														=	50
	i=1, j=1					1			-230													≤	0
	i=1, j=2						1			-130												≤	0
	i=1, j=3							1			-50											≤	0
Bloco 2	i=2, j=1											1	-1			1						=	100
	i=2, j=2												1	-1			1					=	50
	i=2, j=3													1	-1			1				=	80
	i=2, j=1															1			-230			≤	0
	i=2, j=2																1			-130		≤	0
	i=2, j=3																	1			-80	≤	0
Bloco 3	j=1					1			10							1			20			≤	250
	j=2						1			10							1			20		≤	250
	j=3							1			10							1			20	≤	250
	custo	2580	1	1	1	1	1	1	200	200	200	2580	1	1	1	1	1	1	200	200	200		
	solução R. L.	**0**	**0**	**50**	**0**	**100**	**130**	**0**	**0.43**	**1**	**0**	**0**	**0**	**0**	**0**	**100**	**50**	**80**	**0.43**	**0.38**	**1**	**1161**	

Figura 11: Reestruturação da formulação clássica do Exemplo- decomposição por artigo.

O primeiro problema mestre restrito, que se apresenta na Figura 12, baseia-se na utilização da estratégia apresentada na subsecção 4.2.3 para definir um PMR válido, e é neste caso definido por 2 variáveis artificiais, uma por cada artigo e por 2 planos de produção, que correspondem à solução óptima de cada um dos subproblemas, quando se consideram apenas os custos originais. A solução óptima do subproblema relativo ao artigo 1 corresponde a um plano de produção no qual a totalidade da procura desse artigo nos 3 períodos (230 unidades) deve ser produzida no primeiro período. Esta solução apresenta um custo total de 610 u. m.. A solução óptima do subproblema do artigo 2 indica que a totalidade da procura desse artigo (230 unidades) deve também ser produzida no primeiro período, mas neste caso com um custo total de 640 u. m.

	v1	**v2**	**z11**	**z21**		
i=1	1		1		=	1
i=2		1		1	=	1
j=1			240	250	≤	250
j=2					≤	250
j=3					≤	250
custo	2580	2580	610	640		
	0	**0.96**	**1**	**0.04**		

Figura 12: Primeiro problema mestre restrito do Exemplo- decomposição por artigo.

A solução óptima do primeiro PMR tem um custo de 3 112.4 u. m. e corresponde a produzir a totalidade da procura do artigo 1 (230 unidades) no primeiro período de planeamento, sendo toda a procura deste artigo satisfeita e a produzir 9.2 unidades do artigo 2 no primeiro período de planeamento. Relativamente ao artigo 2, apenas 4% da sua procura é satisfeita através de unidades produzidas no período 1, sendo a restante procura, 96%, satisfeita através da variável artificial, que pode ser vista como o inventário existente deste artigo, no início do primeiro período de planeamento. Caso não se tivessem considerado as variáveis artificiais no primeiro PMR, este seria um problema impossível, pelo facto da capacidade disponível no primeiro período não ser suficiente para fazer face à procura total.

A partir da solução óptima do problema mestre restrito, obtêm-se os valores das variáveis duais, necessários para a resolução dos subproblemas. Na Tabela 5, apresentam-se esses valores.

μ_1	-7.76
μ_2	0
μ_3	0
π^1	2472.4
π^2	2580

Tabela 5: Variáveis duais do primeiro PMR – decomposição por artigo.

A solução óptima do subproblema associado ao artigo 1 (após a consideração das variáveis duais do primeiro PMR) é a que se apresenta na Figura 13 e representa um plano de produção que corresponde a produzir 100 unidades do artigo 1 no período 1 para satisfazer a procura desse período e a produzir 130 unidades desse mesmo artigo no período 2 para satisfazer a procura dos períodos 2 e 3. Neste plano, existem portanto 50 unidades que são armazenadas no final do período 2 para satisfazer a procura do período 3. O custo deste plano

de produção é de 680 u. m. e pode ser obtido multiplicando a linha *c1j* apresentada na Figura 13, pela linha onde está representada a solução do subproblema associado ao artigo 1, na mesma figura . Na linha *c1j*, representam-se os custos originais das variáveis de decisão e na linha seguinte os custos associados às variáveis duais $\mu 1j$. A linha denominada por custo modificado representa a subtracção das duas linhas anteriores. O plano de produção que se obtém resolvendo o subproblema, é um plano atractivo e deve ser inserido no problema mestre restrito, pois o valor da função objectivo do subproblema é negativo (-939). O valor da função objectivo obtém-se multiplicando a linha relativa ao custo modificado pela linha seguinte, onde está representada a solução do subproblema e subtraindo a esse resultado o valor da variável dual π^1, já apresentado na Tabela 5.

	s11	**s12**	**s13**	**x11**	**x12**	**x13**	**y11**	**y12**	**y13**		
i=1, j=1	-1			1						=	100
i=1, j=2	1	-1			1					=	80
i=1, j=3		1	-1			1				=	50
i=1, j=1				1			-230			≤	0
i=1, j=2					1			-130		≤	0
i=1, j=3						1			-50	≤	0
c1j	1	1	1	1	1	1	200	200	200		
μ^1j				-7.76	0	0	-77.6	0	0		
custo modificado	1	1	1	8.76	1	1	277.6	200	200		
Solução - Subproblema 1	**0**	**50**	**0**	**100**	**130**	**0**	**1**	**1**	**0**	-939	

Figura 13: Subproblema associado ao artigo 1 – 1ª iteração.

A solução óptima do subproblema associado ao artigo 2 (após a introdução das variáveis duais do primeiro PMR) está representada na Figura 14. O plano de produção que se obtém resolvendo este subproblema é também atractivo, pois o valor da função objectivo do subproblema é negativo e corresponde a produzir 100 unidades do artigo 2 no período 1 para satisfazer a procura desse período e a produzir 130 unidades no período 2 para satisfazer a procura dos períodos 2 e 3, armazenando-se 80 unidades no final do período 2 para satisfazer a procura do período 3. O custo deste plano de produção é de 710 u. m..

	s21	**s22**	**s23**	**x21**	**x22**	**x23**	**y21**	**y22**	**y23**		
i=2, j=1	-1			1						=	100
i=2, j=2	1	-1			1					=	50
i=2, j=3		1	-1			1				=	80
i=2, j=1				1			-230			≤	0
i=2, j=2					1			-130		≤	0
i=2, j=3						1			-80	≤	0
c2j	1	1	1	1	1	1	200	200	200		
μ2j				-7.76	0	0	-155	0	0		
custo modificado	1	1	1	8.76	1	1	355.2	200	200		
Solução - Subproblema 2	**0**	**80**	**0**	**100**	**130**	**0**	**1**	**1**	**0**	-939	

Figura 14: Subproblema associado ao artigo 2 – 1ª iteração.

Uma vez que em ambos os subproblemas existe uma coluna atractiva, o próximo problema mestre restrito será constituído por mais duas variáveis que o primeiro PMR apresentado na Figura 12. Na Figura 15 apresenta-se o PMR e a correspondente solução óptima, após a introdução dos novos planos de produção. O valor da função objectivo deste PMR é de cerca de 1485 u. m..

	v1	**v2**	**z11**	**z12**	**z21**	**z22**		
i=1	1		1	1			=	1
i=2		1			1	1	=	1
j=1			240	110	250	120	≤	250
j=2				140		150	≤	250
j=3							≤	250
custo	2580	2580	610	680	640	710		
	0	**0.06**	**0**	**1**	**0.21**	**0.73**		

Figura 15: PMR do Exemplo após a 1ª iteração do método de geração de colunas.

Com base nos valores das variáveis duais, obtidos a partir da solução óptima do PMR da Figura 15, os subproblema são novamente resolvidos e o processo de geração de colunas prossegue, enquanto existirem planos de produção atractivos. Quando não existirem mais planos de produção atractivos, a solução óptima da relaxação linear da decomposição por artigo foi encontrada. O valor da solução óptima da relaxação linear da decomposição por artigo do Exemplo tem um custo total de 1378 u. m.. Na Figura 16 apresenta-se a solução correspondente.

	v1	v2	z11	z12	z13	z21	z22	z23		
i=1	1		1	1	1				=	1
i=2		1				1	1	1	=	1
j=1			240	110	190	250	120	170	≤	250
j=2				140			150		≤	250
j=3					60			100	≤	250
custo	2580	2580	610	680	710	640	710	680		
	0	**0**	**0**	**1**	**0**	**0**	**0.6**	**0.4**		

Figura 16: Quadro óptimo da relaxação linear da decomposição por artigo.

A partir da solução óptima do PMR apresentada na Figura 16 e das expressões *(4.4)* e *(4.5)* apresentadas na subsecção 4.2.1, pode recuperar-se a solução em termos das variáveis originais, que é a que se apresenta na Tabela 6.

x11	100	**x21**	120
x12	130	**x22**	78
x13	0	**x23**	32
y11	1	**y21**	1
y12	1	**y22**	0.6
y13	0	**y23**	0.4

Tabela 6: – Solução óptima da relaxação linear da decomposição por artigo (em termos das variáveis originais).

Comparando o valor da solução óptima da relaxação linear do Exemplo (1161 u. m.) com o valor da solução da relaxação linear da decomposição por artigo do mesmo exemplo (1378 u. m.), verifica-se que o limite inferior obtido com a decomposição por artigo é de melhor qualidade, pois se encontra mais próximo do valor da solução óptima inteira do Exemplo, que é de 1510 u. m., tal como apresentado na secção 2.3.4.

Analisando a Tabela 6, verifica-se que existem duas variáveis de preparação, y^2_2 e y^2_3, que são fraccionárias, pelo que esta solução não corresponde à solução óptima inteira do Exemplo. Para encontrar a sua solução óptima inteira, utiliza-se o método de partição e geração de colunas.

Neste exemplo escolhe-se a variável y^2_2 como variável de partição. As duas restrições de partição a adicionar ao problema, que darão origem a dois novos nodos na árvore de pesquisa (nodos 1 e 2) são as seguintes:

$$\sum_{k \in \overline{H}} \beta_{2k}^2 z_k^2 = 0 \qquad \text{(nodo 1)}$$

$$\sum_{k \in \overline{H}} \beta_{2k}^2 z_k^2 = 1 \qquad \text{(nodo 2)}$$

sendo $\overline{H}$ igual a seis, pois no PMR actual existem seis planos de produção.

Na Figura 17 apresenta-se o problema a resolver no nodo 2, que é definido pelo PMR actual e pela nova restrição de partição onde se força a variável y^2_2 a tomar o valor 1.

	v1	**v2**	**z11**	**z12**	**z13**	**z21**	**z22**	**z23**		
i=1	1		1	1	1				=	1
i=2		1				1	1	1	=	1
j=1			240	110	190	250	120	170	≤	250
j=2				140			150		≤	250
j=3					60			100	≤	250
y^2_2							1		=	1
custo	2580	2580	610	680	710	640	710	680		
	0.00	**0.00**	**0.00**	**1.00**	**0.00**	**0.00**	**0.60**	**0.40**		

Figura 17: Quadro relativo ao PMR do nodo 2 - decomposição por artigo.

Face à introdução da restrição de partição no PMR do nodo 2, no coeficiente da função objectivo da variável y^2_2, do subproblema associado ao artigo 2, será necessário subtrair o valor da variável dual relacionada com esta nova restrição, para que o custo reduzido deste subproblema seja correctamente determinado.

Após a optimização do nodo 2 através do método de geração de colunas, obtém-se o quadro apresentado na Figura 18, cujo valor da função objectivo é de 1405 u. m.. Comparativamente com o quadro óptimo da relaxação linear da decomposição por artigo apresentado na Figura 16, o quadro da Figura 18 tem mais uma restrição e duas colunas, correspondentes aos planos de produção z^1_4 e z^2_4.

	v1	**v2**	**z11**	**z12**	**z13**	**z14**	**z21**	**z22**	**z23**	**z24**		
i=1	1		1	1	1	1					=	1
i=2		1					1	1	1	1	=	1
j=1			240	110	190	110	250	120	170	120	≤	250
j=2				140		90		150		70	≤	250
j=3					60	60			100	100	≤	250
y^2_2								1		1	=	1
custo	2580	2580	610	680	710	830	640	710	680	830		
	0.00	**0.00**	**0.00**	**0.75**	**0.25**	**0.00**	**0.00**	**0.94**	**0.00**	**0.06**		

Figura 18: Quadro óptimo do nodo 2 - decomposição por artigo.

A solução óptima deste nodo em termos das variáveis originais, determinada a partir da solução óptima do PMR apresentada na figura anterior e das expressões *(4.4)* e *(4.5)*, é a que se apresenta na Tabela 7.

x11	120.0	**x21**	100.0
x12	97.5	**x22**	125.0
x13	12.5	**x23**	5.0
y11	1.0	**y21**	1.0
y12	0.8	**y22**	1.0
y13	0.2	**y23**	0.1

Tabela 7: Solução óptima do nodo 2 em termos das variáveis originais - decomposição por artigo.

Como nesta solução ainda existem variáveis de preparação com valores fraccionários, deverá ocorrer uma nova partição, sendo criados dois novos nodos na árvore de pesquisa.

A pesquisa da solução óptima inteira do Exemplo, prossegue através da aplicação do método de partição e geração de colunas, até que exista a garantia de que a melhor solução inteira foi encontrada. Em cada nodo da árvore de pesquisa é adicionada uma restrição de partição, da mesma forma que a que foi agora descrita, sendo esse novo problema optimizado através do método de geração de colunas. A solução óptima inteira do Exemplo coincide com a solução óptima da formulação clássica, já apresentada na subsecção 2.3.4 e tem um valor de 1510 u. m..

4.3 Decomposição por Período

Na secção 4.2 foi definida uma decomposição para o problema LPMACC, baseada na decomposição das restrições de capacidade. Nesta secção, apresenta-se uma decomposição baseada na decomposição das restrições de equilíbrio de stocks. No nosso melhor conhecimento, esta decomposição foi apenas estudada por (Diaby, Bahl et al. 1992a), que utilizaram a relaxação lagrangeana e resolveram o problema reformulado pelo método do subgradiente. Neste trabalho, aplica-se a decomposição de Dantzig-Wolfe e a resolução do problema reformulado é feita pelo método de geração de colunas.

Se, na formulação clássica do problema LPMACC, se deixar de considerar as restrições de equilíbrio de stocks, o problema decompõe-se num conjunto de problemas independentes por período. Assim sendo, uma decomposição alternativa à decomposição por artigo é a que se baseia na decomposição das restrições de equilíbrio de stocks, que será doravante designada por decomposição por período. Nesta decomposição, as restrições de equilíbrio de stocks (equações *(2.1)* e *(2.2)*) da formulação clássica apresentada na página 16 desta dissertação definem as restrições de ligação que ficam no problema mestre, e as restrições de capacidade *(2.3)*, as restrições que relacionam as variáveis de preparação com as variáveis de produção *(2.4)* e as restrições *(2.7)* e *(2.8)* definem os subproblemas. O número de subproblemas que se obtêm quando se aplica a DDW à formulação clássica é igual ao número de períodos de planeamento.

Na decomposição por período, os pontos extremos dos subproblemas representam padrões de produção que estão associados a um período. Cada padrão de produção indica para um determinado período a quantidade que deve ser produzida do "mix" de artigos a processar.

4.3.1 Definição do Problema Mestre

Para a definição do problema mestre, considerem-se os parâmetros e variáveis definidos na secção 2.3.2, relativos à formulação clássica e o seguinte conjunto, parâmetros e variável de decisão:

Conjunto:

H representa o conjunto de padrões de produção indexados por $k=1,...,t$

Parâmetros:

β^i_{jk} assume o valor *1* caso exista uma preparação no período *j* do artigo *i*, no padrão de produção *k* e assume o valor *0* caso contrário; *i=1,...,n ; j=1,...,m ; k=1,...t*

α^i_{jk} representa a quantidade a produzir no período *j* do artigo *i*, no padrão de produção *k; i=1,...,n ; j=1,...,m ; k=1,...t*

Variável de Decisão:

z_{jk} representa o peso do padrão de produção *k* do período *j; j=1,...,m ; k=1,...t*

A solução da relaxação linear do problema LPMACC, quando se aplica a decomposição por período, pode ser obtida resolvendo o seguinte problema mestre, aqui designado por *PMJ*:

$$Z_{PMJ} = Min \sum_{i=1}^{n}\sum_{j=1}^{m} h_j^i s_j^i + \sum_{j=1}^{m}\sum_{k=1}^{t}\left[\left(\sum_{i=1}^{n} q_j^i \beta_{jk}^i + p_j^i \alpha_{jk}^i\right) z_{jk}\right] \qquad (PMJ)$$

sujeito a:

$$\sum_{k=1}^{t} z_{jk} \le 1 \quad \forall j \in J \quad (\pi_j) \qquad (4.13)$$

$$\sum_{k=1}^{t} \alpha_{1k}^i z_{1k} = d_1^i + s_1^i \quad \forall i \in I \quad (\mu_1^i) \qquad (4.14)$$

$$s_{j-1}^i + \sum_{k=1}^{t} \alpha_{jk}^i z_{jk} = d_j^i + s_j^i \quad \forall i \in I, \forall j \in J \setminus \{1\} \quad (\mu_j^i) \qquad (4.15)$$

$$s_m^i = 0 \quad \forall i \in I \qquad (4.16)$$

$$s_j^i \ge 0 \quad \forall i \in I, \forall j \in J \setminus \{m\} \qquad (4.17)$$

$$z_{jk} \ge 0 \quad \forall j \in J, \forall k \in H\,. \qquad (4.18)$$

As variáveis de decisão z_{jk} podem ser vistas como a proporção da procura do período *j* que é satisfeita pelo padrão de produção *k*. A função objectivo do problema mestre minimiza a soma dos custos totais. As restrições *(4.13)* representam as restrições de convexidade. Existe uma restrição de convexidade associada a cada período na qual se força a escolha de uma combinação de padrões de produção. Associada a cada restrição de convexidade está uma variável dual π_j. No caso da decomposição por período, as restrições de convexidade podem ser escritas como uma desigualdade do tipo ≤, em vez da sua forma habitual, na qual estas restrições são escritas como uma igualdade. Esta alteração é possível, devido ao espaço de soluções admissíveis do subproblema *SPJ* (que será definido na secção seguinte) incluir a origem. Com esta modificação, excluem-se do problema mestre os padrões de produção que

indiquem que para um determinado período nada deve ser produzido.

O conjunto de restrições *(4.14)* e *(4.15)*, são as restrições de equilíbrio de stocks. Neste conjunto força-se a que a combinação de padrões de produção escolhida respeite a procura de cada artigo em cada período. Este conjunto tem associadas as variáveis duais μ_j^i. As restrições *(4.16)* forçam o inventário de cada artigo a ser nulo no último período de planeamento e as restrições *(4.17)* impõem a condição de não negatividade das variáveis de inventário. Por último, as restrições *(4.18)* forçam as variáveis de peso a tomarem valores não negativos.

Em termos da dimensão do problema mestre *PMJ*, este é constituído por $J+I\times J$ restrições. Comparando com o número de restrições da formulação clássica, que são $J+2\times I\times J$, onde I representa o número de artigos e J o número de períodos de planeamento, conclui-se que com a decomposição por período o número linhas do problema se reduz significativamente. Tal como na decomposição por artigo, o número de variáveis do problema mestre que resulta da aplicação da decomposição por período é extremamente elevado, pelo que nesta decomposição também se recorre ao método de geração de colunas para resolver o problema *PMJ*.

Para recuperar a solução do problema *C*, a partir da solução óptima do problema mestre *PMJ*, deverão utilizar-se as expressões:

$$x_j^i = \sum_{k=1}^{t} \alpha_{jk}^i z_{jk} \qquad (4.19)$$

$$y_j^i = \sum_{k=1}^{t} \beta_{jk}^i z_{jk} \qquad (4.20)$$

O valor das variáveis s_j^i pode ser lido directamente no problema mestre *PMJ*.

4.3.2 Subproblemas

Aplicando ao problema *C* (definido na subsecção 2.3.2) a decomposição por período, obtêm-se J subproblemas, um por cada período de planeamento. Cada subproblema é definido por:

$$Z_{SPJ} = Min\sum_{i=1}^{n} p_j^i x_j^i + \sum_{i=1}^{n} q_j^i y_j^i - \sum_{i=1}^{n} x_j^i \mu_j^i - \pi_j \qquad (SPJ)$$

sujeito a:

$$\sum_{i=1}^{n} a_j^i x_j^i + \sum_{i=1}^{n} b_j^i y_j^i \le c_j \qquad (4.21)$$

$$x_j^i \le min\left\{\frac{c_j - b_j^i}{a_j^i}, \sum_{t=j}^{m} d_t^i\right\} y_j^i \quad \forall i \in I \qquad (4.22)$$

$$x_j^i \ge 0 \quad \forall i \in I \qquad (4.23)$$

$$y_j^i \in \{0,1\} \ \forall i \in I, \qquad (4.24)$$

onde π_j representa a variável dual associada à restrição *j* do conjunto de restrições de convexidade *(4.13)* e $\mu^i{}_j$ representa o vector de variáveis duais associado ao conjunto de restrições de equilíbrio de stocks (conjunto de restrições *(4.14)* e *(4.15)*).

Na função objectivo do subproblema procura-se o padrão de produção associado ao período *j*, que tem o menor custo reduzido. As restrições que definem o subproblema são, tal como já referido, as restrições de capacidade *(4.21)*, as restrições que garantem que uma preparação ocorre caso no período *j* se produza o artigo *i (4.22)*, as restrições de não negatividade *(4.23)* e as restrições de integralidade *(4.24)*.

Após resolver o subproblema, caso o seu custo reduzido seja menor que zero, existe um padrão de produção que é atractivo e que deve ser adicionado ao problema mestre. Como na decomposição por período, existem *J* subproblemas, o número de novos padrões de produção a adicionar ao problema mestre, em cada iteração do método de geração de colunas, será no máximo igual ao número de períodos.

4.3.3 Primeiro Problema Mestre Restrito

O problema mestre *PMJ* tem um número exponencialmente grande de variáveis, pelo que faz todo o sentido considerarmos neste problema apenas um número restrito do total das variáveis que o definem e utilizar o método de geração de colunas para o resolver. Vimos já que o primeiro problema mestre restrito tem de ser um problema válido para que se possa iniciar o processo iterativo de geração de colunas.

Na decomposição por período, considerando no primeiro PMR apenas as variáveis de inventário, a validade deste problema não fica garantida devido ao conjunto de restrições *(4.14)*.

Para validar o primeiro problema mestre restrito, pode adoptar-se uma estratégia que tem a ver com a adição de um conjunto de variáveis artificiais às restrições de equilíbrio de stocks do primeiro período de planeamento, conjunto *(4.14)*. A cada restrição deste conjunto, associa-se uma variável artificial, v^i, pelo que no total são inseridas I novas variáveis artificiais. As variáveis artificiais deste conjunto podem ser interpretadas como sendo o inventário existente no início do primeiro período de planeamento. As alterações à formulação matemática do problema mestre são as seguintes: no conjunto de restrições de equilíbrio de stocks do primeiro período de planeamento, *(4.14),* deve adicionar-se o termo v^i e na função objectivo do problema *PMJ* deve adicionar-se o termo $\sum_{i=1}^{n} f^i v^i$, onde f^i é um escalar com um valor muito elevado, que torna qualquer solução com variáveis artificiais positivas pior que uma solução óptima em termos das variáveis de peso z_{jk}. Nesta dissertação, é utilizada esta estratégia na definição do primeiro PMR.

Para além destas variáveis, que permitem validar o problema, podem ainda associar-se variáveis artificiais às restrições de convexidade *(4.13)*, adicionando-se ao termo independente de cada restrição de convexidade, uma variável artificial, v_j. Com a introdução destas variáveis aumenta-se a disponibilidade de capacidade em cada período. Na formulação do problema mestre, existem neste caso também algumas alterações na sua estrutura. No conjunto de restrições de convexidade, *(4.13)*, deve adicionar-se ao termo independente de cada restrição a variável v_j, e na função objectivo do problema *PMJ* deve adicionar-se o termo $\sum_{j=1}^{m} f_j v_j$. A constante f_j tem um valor extremamente elevado, que torna qualquer solução com variáveis artificiais positivas pior que uma solução óptima em termos das variáveis de peso z_{jk}.

4.3.4 Regras de Partição

A solução óptima do problema mestre definido na subsecção 4.3.1 corresponde à solução óptima da relaxação linear da decomposição por período. Esta solução representa um limite inferior para o valor do óptimo inteiro, que é melhor ou igual que o obtido pela solução da relaxação linear da formulação clássica, pois os subproblemas que resultam desta decomposição não têm a propriedade da integralidade (Geoffrion 1974).

Aplicando a decomposição por período à formulação clássica do problema *C* e resolvendo o problema reformulado (problema mestre *PMJ*) por geração de colunas, a solução óptima desse problema mestre não corresponde necessariamente à solução óptima do problema *C*. Na subsecção 4.3.1 foi apresentado um procedimento que permite através de uma solução

óptima do problema mestre, determinar a solução correspondente do problema *C*, através do recurso às expressões *(4.19)* e *(4.20)*. Se, nessa solução, existirem variáveis de preparação, as variáveis y^i_j, que não sejam binárias, esta solução não corresponde à solução óptima do problema *C*. Para determinar a sua solução óptima deverá então recorrer-se ao método de partição e geração de colunas, já apresentado na secção 3.5.

Sempre que a solução óptima da raiz da árvore de pesquisa, ou a solução óptima de um qualquer nodo da árvore não for inteira e caso exista interesse em explorar esse nodo, devem adicionar-se ao problema dois novos nodos. Cada um destes nodos passa a definir um "novo" problema, definido pelo PM actual e por uma das restrições de partição abaixo apresentadas:

$$\sum_{k \in \overline{H}} \beta^i_{jk} z_{jk} = 0$$

$$\sum_{k \in \overline{H}} \beta^i_{jk} z_{jk} = 1$$

onde $\overline{H}$ representa o conjunto de padrões de produção que fazem parte do problema mestre actual.

A regra de partição usada nesta decomposição é a que se baseia nas variáveis originais, sendo adicionadas restrições de partição às variáveis de preparação y^i_j que tenham valores fraccionários.

Em cada nodo da árvore de pesquisa, devem efectuar-se os seguintes passos:

- Aplicar o método de geração de colunas;
- Analisar a solução óptima em termos das variáveis originais;
- Se existirem nessa solução, variáveis de preparação com valor fraccionário, adicionar as restrições de partição ao problema mestre e passar para um novo nodo da árvore de pesquisa, senão parar.

Na aplicação do método de geração de colunas em cada nodo da árvore de pesquisa, deverá atender-se que, face à introdução da restrição de partição desse nodo, para que os custos reduzidos dos subproblemas a resolver no processo de geração de colunas sejam correctamente determinados, a expressão do custo reduzido deve ser alterada. Considere-se que τ^i_j representa a variável dual associada à restrição de partição que foi adicionada ao nodo em questão. A expressão do custo reduzido, definida na função objectivo de cada um dos *J* subproblemas da subsecção 4.3.2, deverá integrar o seguinte termo:

$$-\sum_{i=1}^{n} y_j^i \tau_j^i$$

contabilizando-se desta forma a restrição de partição que foi adicionada ao problema mestre .

Tal como na decomposição por artigo, na decomposição por período também não é conveniente utilizar a estratégia de partição baseada nas variáveis de peso, definida na secção 3.5, pelos motivos já mencionados na subsecção 4.2.4.

4.3.5 Exemplo

Na subsecção 4.2.5, para exemplificar a aplicação da decomposição por artigo foi apresentada a resolução do Exemplo definido na subsecção 2.3.4 desta dissertação. Nesta subsecção, o referido exemplo será novamente resolvido, agora através da decomposição por período.

Antes de aplicar a decomposição por período à formulação clássica do Exemplo, apresenta-se na Figura 19 uma reestruturação da formulação clássica desse exemplo, apresentada na Figura 5 da página 20. Esta reestruturação consiste na reordenação das variáveis e restrições do problema, por forma a tornar clara a existência de uma estrutura angular em blocos, alternativa à já apresentada na decomposição por artigo.

		s11	**s21**	**s12**	**s22**	**s13**	**s23**	**x11**	**x21**	**y11**	**y21**	**x12**	**x22**	**y12**	**y22**	**x13**	**x23**	**y13**	**y23**		
Bloco 1	j=1							1	1	10	20									≤	250
	i=1, j=1							1		-230										≤	0
	i=2, j=1								1		-230									≤	0
Bloco 2	j=2											1	1	10	20					≤	250
	i=1, j=2											1		-130						≤	0
	i=2, j=2												1		-130					≤	0
Bloco 3	j=3															1	1	10	20	≤	250
	i=1, j=3															1		-50		≤	0
	i=2, j=3																1		-80	≤	0
Bloco 4	i=1, j=1	-1						1												=	100
	i=2, j=1		-1						1											=	100
	i=1, j=2	1		-1								1								=	80
	i=2, j=2		1		-1								1							=	50
	i=1, j=3			1		-1										1				=	50
	i=2, j=3				1		-1										1			=	80
	custo	1	1	1	1	1	1	1	1	200	200	1	1	200	200	1	1	200	200		
	solução R. L.	**0**	**0**	**50**	**0**	**0**	**0**	**100**	**100**	**0.43**	**0.43**	**130**	**50**	**1**	**0.38**	**0**	**80**	**0**	**1**	1161	

Figura 19: Reestruturação da formulação clássica do Exemplo– decomposição por período.

A partir desta reestruturação, torna-se evidente a existência de um conjunto de restrições que está associado aos três períodos de planeamento (bloco 4) e de outros três conjuntos de restrições que têm a mesma estrutura (diferindo apenas no período a que dizem respeito) e que estão associados a apenas um período de planeamento (blocos 1, 2 e 3).

Com base nesta estrutura, definem-se três subproblemas e um problema mestre. As restrições do bloco 4 ficam no problema mestre. As restrições do bloco 1 definem o subproblema 1, as do bloco 2, definem o subproblema 2 e por último as restrições do bloco 3, definem o subproblema 3.

Para definir o primeiro problema mestre restrito, que se apresenta na Figura 20, foi utilizada a estratégia exposta na subsecção 4.3.3, para definir um PMR válido, que se baseia na utilização de duas variáveis artificiais, cada uma delas associada à restrição de equilíbrio de stocks do artigo *i*, no primeiro período de planeamento.

	v1	**v2**	**s11**	**s21**	**s12**	**s22**	**s13**	**s23**		
j=1									≤	1
j=2									≤	1
j=3									≤	1
i=1, j=1	1		-1						=	100
i=2, j=1		1		-1					=	100
i=1, j=2			1		-1				=	80
i=2, j=2				1		-1			=	50
i=1, j=3					1		-1		=	50
i=2, j=3						1		-1	=	80
custo	2580	2580	1	1	1	1	1	1		
	230	**230**	**130**	**130**	**50**	**80**	**0**	**0**		

Figura 20: Primeiro problema mestre restrito do Exemplo– decomposição por período.

A solução óptima do primeiro PMR tem um custo de 1 187 190 unidades monetárias, e corresponde a satisfazer toda a procura através das variáveis artificiais, que como já referido anteriormente podem ser interpretadas como sendo o inventário existente no início do primeiro período de planeamento.

A partir da solução óptima do problema mestre restrito, faz-se a leitura dos valores das variáveis duais associadas às restrições de convexidade e às restrições de equilíbrio de stocks e resolvem-se os subproblemas. Na Tabela 8 apresentam-se os valores dessas variáveis duais.

π_1	0
π_2	0
π_3	0
μ^1_1	2580
μ^2_1	2580
μ^1_2	2581
μ^2_2	2581
μ^1_3	2582
μ^2_3	2582

Tabela 8: Variáveis duais do primeiro PMR – decomposição por período.

A solução óptima do subproblema associado ao primeiro período de planeamento apresenta-se na Figura 21. Esta solução corresponde a um padrão de produção, no qual a totalidade da procura do artigo 1 (230 unidades) é produzida no período 1. Face à restrição de capacidade do período 1, que indica que estão disponíveis 250 unidades de tempo, não é possível nesse mesmo período produzir o produto 2. Como na preparação do artigo 1 são gastas 10 unidades de tempo, e por cada unidade produzida se gasta 1 unidade de tempo, na produção das 230 unidades, gastam-se 240 unidades de tempo. Existem portanto 10 unidades de tempo que ficam disponíveis, mas que não permitem a produção de qualquer unidade do artigo 2, pois só para a sua preparação seriam necessárias 20 unidades de tempo. O custo total deste padrão de produção é de 430 u. m.. O valor do custo total deste padrão pode ser obtido através da multiplicação da linha *ci1*, apresentada na figura em baixo, com a linha onde está representada a solução óptima do subproblema 1 na mesma figura. A linha *ci1* diz respeito aos custos originais das variáveis de decisão que fazem parte do subproblema associado ao primeiro período. Na linha $\mu i1$ representam-se os valores das variáveis duais das restrições de equilíbrio de stocks associadas ao primeiro período. A linha denominada por custo modificado resulta da subtracção da linha *ci1* pela linha $\mu i1$ e representa os coeficientes da função objectivo do subproblema 1.

O valor da solução óptima do subproblema 1 é igual a -592 970 u. m., pelo que o padrão de produção obtido nesta solução é atractivo e deve ser inserido no primeiro PMR. Para determinar o valor da solução óptima do subproblema 1 deve multiplicar-se a linha do custo modificado pela linha seguinte, onde está representada a solução óptima do subproblema 1, e a esse resultado deve subtrair-se o valor da variável dual associada à restrição de convexidade do período 1 (π_1), apresentado na Tabela 8.

	x11	x21	y11	y21		
j=1	1	1	10	20	≤	250
i=1, j=1	1		-230		≤	0
i=2, j=1		1		-230	≤	0
ci1	1	1	200	200		
μi1	2580	2580				
custo modificado	-2579	-2579	200	200		
Solução - Subproblema 1	**230**	**0**	**1**	**0**	-592970	

Figura 21: Subproblema associado ao período 1 – 1ª iteração.

Na Figura 22 apresenta-se a solução óptima do subproblema associado ao 2º período de planeamento, que corresponde a um padrão de produção que tem um custo total de 620 u. m., no qual se produzem 90 unidades do artigo 1 e 130 unidades do artigo 2, no período 2. Este padrão de produção é também atractivo, porque o valor da solução óptima deste subproblema é menor que zero, como se pode verificar na figura.

	x12	x22	y12	y22		
j=2	1	1	10	20	≤	250
i=1, j=2	1		-130		≤	0
i=2, j=2		1		-130	≤	0
ci2	1	1	200	200		
μi2	2581	2581				
custo modificado	-2580	-2580	200	200		
Solução - Subproblema 2	**90**	**130**	**1**	**1**	-567200	

Figura 22: Subproblema associado ao período 2 – 1ª iteração.

Por último, na Figura 23 representa-se a solução óptima do subproblema associado ao 3º período de planeamento. O padrão de produção do 3ª período indica que devem ser produzidas 50 unidades do artigo 1 e 80 unidades do artigo 2 nesse mesmo período e tem um custo total de 530 u. m.. Este padrão de produção deve ser inserido no primeiro PMR, pois o valor do seu custo reduzido é negativo.

	x13	x23	y13	y23		
j=3	1	1	10	20	≤	250
i=1, j=3	1		-50		≤	0
i=2, j=3		1		-80	≤	0
ci3	1	1	200	200		
μi3	2582	2582				
custo modificado	-2581	-2581	200	200		
Solução - Subproblema 3	**50**	**80**	**1**	**1**	-335130	

Figura 23: Subproblema associado ao período 3 – 1ª iteração.

Como os três padrões de produção obtidos na resolução dos subproblemas são atractivos, devem ser inseridos no primeiro PMR. O próximo PMR será constituído pelo primeiro PMR já apresentado na Figura 20 e por mais três novas colunas, respeitantes aos três padrões de produção atractivos. Na Figura 24 apresenta-se esse problema mestre restrito e a sua solução óptima. O valor da função objectivo deste PMR é igual a 259 052 u. m..

	v1	v2	s11	s21	s12	s22	s13	s23	z11	z21	z31		
j=1									1			≤	1
j=2										1		≤	1
j=3											1	≤	1
i=1, j=1	1		-1						230			=	100
i=2, j=1		1		-1								=	100
i=1, j=2			1		-1					90		=	80
i=2, j=2				1		-1				130		=	50
i=1, j=3					1		-1				50	=	50
i=2, j=3						1		-1			80	=	80
custo	2580	2580	1	1	1	1	1	1	430	620	530		
	0	**100**	**40**	**0**	**50**	**80**	**0**	**0**	**0.61**	**1**	**0**		

Figura 24: PMR do Exemplo após a 1ª iteração do método de geração de colunas.

Com base nos valores das variáveis duais, obtidos a partir da solução óptima do PMR da Figura 24, os subproblemas são novamente resolvidos e o processo de geração de colunas prossegue, enquanto existirem padrões de produção atractivos. Quando não existirem mais padrões de produção atractivos, pode dizer-se que a solução óptima da relaxação linear da decomposição por período foi encontrada. O valor da solução óptima da relaxação linear da decomposição por período do Exemplo tem um custo total de 1184 u. m.. Na Figura 25 apresenta-se a solução correspondente.

	v1	v2	s11	s21	s12	s22	s13	s23	z11	z12	z21	z22	z23	z31	z32	z33		
j=1									1	1							≤	1
j=2											1	1	1				≤	1
j=3														1	1	1	≤	1
i=1, j=1	1		-1						230								=	100
i=2, j=1		1		-1						230							=	100
i=1, j=2			1		-1						90		130				=	80
i=2, j=2				1		-1					130	130					=	50
i=1, j=3					1		-1							50		50	=	50
i=2, j=3						1		-1						80	80		=	80
custo	2580	2580	1	1	1	1	1	1	430	430	620	330	330	530	280	250		
	0	**0**	**30**	**0**	**50**	**0**	**0**	**0**	**0.57**	**0.43**	**0.22**	**0.16**	**0.62**	**0**	**1**	**0**		

Figura 25: Quadro óptimo da relaxação linear da decomposição por período.

Através da solução óptima da relaxação linear da decomposição por período e das expressões *(4.19)* e *(4.20)* apresentadas na página 59 recupera-se a solução em termos das variáveis originais. Na Tabela 9 pode ler-se essa solução.

x11	130	**x21**	100
x12	100	**x22**	50
x13	0	**x23**	80
y11	0.57	**y21**	0.43
y12	0.84	**y22**	0.38
y13	0	**y23**	1

Tabela 9 : Solução óptima da relaxação linear da decomposição por período (em termos das variáveis originais).

A solução apresentada no quadro anterior não corresponde à solução óptima inteira do problema, pois nesta solução existem várias variáveis de preparação que têm valores fraccionários. Para determinar a solução óptima inteira do Exemplo, vamos utilizar o método de partição e geração de colunas.

Para fazer a partição escolhe-se a variável y^2_1. As novas restrições a adicionar ao problema são as seguintes:

$$\sum_{k \in \overline{H}} \beta^2_{1k} z_{1k} = 0 \qquad \text{(nodo 1)}$$

$$\sum_{k \in \overline{H}} \beta^2_{1k} z_{1k} = 1 \qquad \text{(nodo 2)}$$

sendo $\overline{H}$ igual a oito, pois no PMR actual existem oito padrões de produção.

Na Figura 26 apresenta-se o problema mestre restrito a resolver no nodo 2, que é constituído pelo PMR actual e pela restrição de partição associada ao nodo 2.

	v1	**v2**	**s11**	**s21**	**s12**	**s22**	**s13**	**s23**	**z11**	**z12**	**z21**	**z22**	**z23**	**z31**	**z32**	**z33**		
j=1									1	1							≤	1
j=2											1	1	1				≤	1
j=3														1	1	1	≤	1
i=1, j=1	1		-1						230								=	100
i=2, j=1		1		-1						230							=	100
i=1, j=2			1		-1						90		130				=	80
i=2, j=2				1		-1					130	130					=	50
i=1, j=3					1		-1							50		50	=	50
i=2, j=3						1		-1						80	80		=	80
y^2_1										1							=	1
custo	2580	2580	1	1	1	1	1	1	430	430	620	330	330	530	280	250		
	0	**0**	**30**	**0**	**50**	**0**	**0**	**0**	**0.57**	**0.43**	**0.22**	**0.16**	**0.62**	**0**	**1**	**0**		

Figura 26: Quadro relativo ao PMR do nodo 2 – decomposição por período.

Para que após a adição da restrição de partição o custo do subproblema 1 seja correctamente determinado, no coeficiente da função objectivo associado a y^2_1, deve subtrair-se o valor da variável dual associada à restrição de partição.

Na Figura 27 representa-se a solução óptima do nodo 2. O custo total desta solução é de aproximadamente 1281 u. m..

	v1	**v2**	**s11**	**s21**	**s12**	**s22**	**s13**	**s23**	**z11**	**z12**	**z13**	**z21**	**z22**	**z23**	**z24**	**z31**	**z32**	**z33**		
j=1									1	1	1								≤	1
j=2												1	1	1	1				≤	1
j=3																1	1	1	≤	1
i=1, j=1	1		-1						230		220								=	100
i=2, j=1		1		-1						230	0								=	100
i=1, j=2			1		-1							90		130	130				=	80
i=2, j=2				1		-1						130	130		90				=	50
i=1, j=3					1		-1									50		50	=	50
i=2, j=3						1		-1								80	80		=	80
y^2_1										1	1								=	1
custo	2580	2580	1	1	1	1	1	1	430	430	620	620	330	330	620	530	280	250		
	0	**0**	**0**	**25**	**50**	**0**	**0**	**0**	**0**	**0.55**	**0.45**	**0**	**0**	**0.73**	**0.27**	**0**	**1**	**0**		

Figura 27: Quadro óptimo do nodo 2 – decomposição por período.

Em termos das variáveis originais a solução óptima do nodo 2 corresponde a:

x11	100	**x21**	125
x12	130	**x22**	25
x13	0	**x23**	80
y11	0.45	**y21**	1
y12	1	**y22**	0.27
y13	0	**y23**	1

Tabela 10: Solução óptima do nodo 2 em termos das variáveis originais – decomposição por período.

Como nesta solução ainda existem duas variáveis de preparação com valores fraccionários, deverá ocorrer uma nova partição, sendo criados dois novos nodos na árvore de pesquisa.

A pesquisa da solução óptima inteira do Exemplo, prossegue através da aplicação do método de partição e geração de colunas, até que exista a garantia de que a melhor solução inteira foi encontrada. Em cada nodo da árvore de pesquisa é adicionada uma restrição de partição, da mesma forma que a que foi agora descrita, sendo esse novo problema optimizado através do método de geração de colunas. A solução óptima inteira do Exemplo, coincide com a solução óptima da formulação clássica, já apresentada na subsecção 2.3.4.

5 Implementação e Testes Computacionais

Neste capítulo analisam-se os aspectos relacionados com a implementação dos dois modelos de decomposição estudados nesta dissertação (decomposição por artigo e decomposição por período) e apresentam-se alguns testes computacionais. Nos testes computacionais, a partir da resolução de um conjunto de instâncias para o problema LPMACC que foram retiradas da literatura, comparam-se os resultados obtidos com a decomposição por artigo e com a decomposição por período, com os resultados obtidos usando um "solver" de problemas de programação inteira mista, o CPLEX.

Na secção 5.1 apresenta-se o ADDing e abordam-se os aspectos relacionados com a implementação do modelo da decomposição por artigo e do modelo da decomposição por período.

Na secção 5.2 descrevem-se resumidamente as instâncias de teste escolhidas para a realização dos testes computacionais, analisam-se e comparam-se os resultados da relaxação linear e analisam-se e comparam-se os resultados obtidos na resolução do problema inteiro.

5.1 Implementação

Para resolver as duas decomposições de Dantzig-Wolfe que são objecto desta dissertação e que foram já apresentadas no capítulo 4, utilizou-se o software ADDing ("Automatic Dantzig-Wolfe Decomposition for Integer Column Generation") (Alvelos 2005).

Este software implementa um algoritmo de partição e geração de colunas, que pode ser usado na resolução de problemas de programação inteira pura ou mista, quando se utilizam métodos de decomposição. Em (Alvelos 2005) é feita uma descrição pormenorizada deste software e dos seus modos de utilização.

O ADDing é constituído por um conjunto de classes de C++ e pode ser usado segundo duas perspectivas: como uma "caixa preta" ou como uma "caixa de ferramentas". Na primeira perspectiva, que é a que foi usada neste trabalho, o utilizador tem apenas de especificar a formulação do problema original de programação inteira mista ou pura e o modelo de decomposição que pretende usar, não tendo de se preocupar com a implementação do algoritmo de partição e geração de colunas. Quando usado como uma "caixa de ferramentas", para além da especificação do problema, fica também a cargo do utilizador a implementação do subproblema e/ou do esquema de partição. Deste modo permite-se ao utilizador aumentar a eficiência do algoritmo, através da definição de um subproblema e de regras de partição que têm em conta a estrutura do problema.

Para que se possa usar o ADDing como uma "caixa-preta", o utilizador tem de definir uma classe de C++ que funciona como um dos "inputs" do programa. Essa classe deriva da classe denominada de Decmodel e é nessa classe que o utilizador define a formulação do seu problema original e o modelo de decomposição que pretende usar. O outro "input" do programa é um ficheiro de parâmetros onde o utilizador pode definir/escolher um conjunto de opções relacionadas com várias características do algoritmo de partição e geração de colunas. Como "output", o programa gera um ficheiro de resultados, que contém informações acerca dos resultados obtidos na optimização do problema e informações relacionadas com os parâmetros utilizados na sua optimização.

Com a perspectiva de usar o ADDing para resolver o problema de lotes de produção multi-artigo capacitado com tempos de preparação, implementaram-se dois modelos de

decomposição, um associado à decomposição por artigo e outro associado à decomposição por período. Nos parágrafos seguintes, abordam-se os aspectos relacionados com a definição dos dois modelos de decomposição.

Para implementar o modelo da decomposição por artigo definiu-se um classe de C++, denominada por "instance_ls", onde se faz a leitura de uma instância do problema LPMACC e definiu-se outra classe de C++, denominada por "ls_lot_mip", na qual se construiu o modelo da decomposição por artigo com base num conjunto de funções da classe Decmodel. A classe "instance_ls" é usada pela classe "ls_lot_mip" na definição do modelo. O código relativo às classes "instance_ls" e "ls_lot_mip" apresenta-se em anexo.

O conjunto de funções da classe Decmodel que se implementaram na classe "ls_lot_mip", permitiram definir o problema mestre, os subproblemas e o primeiro problema mestre restrito da decomposição por artigo.

Nesta decomposição, o problema mestre é definido pelas restrições de capacidade e pelas restrições de convexidade. Na fase da implementação alterou-se o sentido das restrições de convexidade da igualdade para uma desigualdade do tipo $\geq$, já que o modelo que se obtém é equivalente ao modelo no qual estas restrições são do tipo = e ao alterarmos o sentido das restrições de convexidade para o tipo $\geq$, o espaço dual é restringido (variáveis duais associadas às restrições de convexidade deixam de ser livres e passam a ser do tipo $\geq$), o que se traduz num benefício para o processo de geração de colunas (Carvalho 2000). Note-se que, numa solução óptima, todas as restrições de convexidade são satisfeitas na igualdade, caso contrário, a quantidade produzida seria maior do que a procura, o que se traduziria num custo adicional desnecessário.

Para definir um primeiro problema mestre restrito válido, adoptou-se a estratégia definida na subsecção 4.2.3. A cada restrição de convexidade adicionou-se uma variável artificial com um coeficiente elevado na função objectivo, que pode ser interpretado como um custo artificial. Para determinar o valor do custo artificial utilizou-se a seguinte expressão: (procura total $\times$ número de artigos $\times$ número de períodos $\times$ custo máximo de armazenagem) + (número de artigos $\times$ número de períodos $\times$ custo máximo de preparação).

Os problemas mestres, quer na raiz da árvore de pesquisa, quer em qualquer nodo da árvore, são resolvidos até à optimalidade pelo algoritmo dual do simplex, através do CPLEX (ILOG 2002) e no método de partição e geração de colunas utiliza-se uma estratégia de partição baseada nas variáveis originais (variáveis de preparação no caso do problema LPMACC), que

está incorporada no ADDing. Os subproblemas da decomposição por artigo são também resolvidos até à optimalidade, pelo algoritmo dual do simplex através do CPLEX.

Embora o ADDing permita fazer a remoção de colunas e/ou inserção/remoção de restrições de convexidade ou de restrições de ligação no problema mestre, de acordo com determinados critérios, nesta decomposição estas potencialidades não foram experimentadas.

Para implementar o modelo da decomposição por período definiu-se um classe de C++, denominada por "ls_knps_mip", onde a partir de um conjunto de funções da classe Decmodel, se definiu o modelo dessa decomposição. A classe "instance_ls", já explicada aquando da apresentação do modelo de decomposição baseado na decomposição por artigo, é também usada pela classe "ls_knps_mip". O código relativo à classe "ls_knps_mip" apresenta-se em anexo. Através desta classe, definiram-se o problema mestre, os subproblemas e o primeiro problema mestre restrito da decomposição por período.

Na decomposição por período, como vimos já na secção 4.3, o problema mestre é definido pelas restrições de equilíbrio de stocks e pelas restrições de convexidade, estando o número de restrições de convexidade associado ao número de períodos. Com a mesma justificação que foi acima referida em relação às restrições de convexidade na decomposição por artigo, na fase da implementação da decomposição por período alterou-se o sentido das restrições de equilíbrio de stocks para o tipo $\geq$, com o objectivo de restringir o espaço dual e melhorar o processo de geração de colunas. Note-se que, tal como na outra decomposição, o modelo alterado é equivalente ao modelo no qual estas restrições são expressas como igualdades.

Na definição do primeiro problema mestre restrito, para validar esse problema foi adoptada a estratégia definida na subsecção 4.3.3, que consiste na adição de um conjunto de variáveis artificiais às restrições de equilíbrio de stocks do primeiro período de planeamento. Estas variáveis artificiais têm associado um coeficiente elevado na função objectivo, que aqui se denomina de custo artificial, que torna qualquer solução em termos das variáveis artificiais pior que uma solução óptima em termos das restantes variáveis. O custo artificial associado a cada variável artificial é tal como na decomposição por artigo, calculado através da seguinte expressão: (procura total $\times$ número de artigos $\times$ número de períodos $\times$ custo máximo de armazenagem) + (número de artigos $\times$ número de períodos $\times$ custo máximo de preparação).

Quer na raiz da árvore de pesquisa, quer em qualquer nodo da árvore, os problemas mestres são resolvidos pelo algoritmo dual do simplex, através de uma função do CPLEX

(ILOG 2002) que tem implementado este algoritmo. Nas primeiras iterações do método de geração de colunas, os problemas mestres da decomposição por período não são resolvidos até à optimalidade. Com este procedimento, pretende-se gerar nas primeiras iterações do método um número elevado de colunas por forma a encontrar rapidamente soluções admissíveis e só após um determinado número de iterações se começam a resolver os problemas mestres até à optimalidade.

Os subproblemas da decomposição por período são resolvidos até à optimalidade, pelo algoritmo dual do simplex, através do CPLEX e no método de partição e geração de colunas, a estratégia de partição usada foi também a que se baseia nas variáveis originais.

Tal como na decomposição por artigo, na decomposição por período também não se fez a remoção de colunas e a remoção/inserção das restrições de convexidade e de ligação no problema mestre.

Quer na decomposição por artigo, quer na decomposição por período, testaram-se duas estratégias de pesquisa da árvore. Na primeira estratégia utilizou-se a pesquisa primeiro em profundidade e deu-se prioridade ao nodo de pesquisa que tem associada a restrição de partição do tipo $\geq$ e na regra de partição utilizou-se uma estratégia baseada na escolha da variável com parte fraccionária mais próxima de *1*. Na segunda estratégia de pesquisa da árvore, também se utilizou a pesquisa primeiro em profundidade, mas deu-se prioridade ao nodo de pesquisa que tem associada a restrição de partição do tipo $\leq$ e na regra de partição utilizou-se uma estratégia baseada na escolha da variável com parte fraccionária mais próxima de *0*.

5.2 Testes Computacionais

A partir de um conjunto de 10 instâncias do problema LPMACC, realizaram-se alguns testes computacionais com o objectivo de avaliar o desempenho quer da decomposição por artigo, quer da decomposição por período.

Os testes computacionais foram realizados num computador pessoal com um processador Intel Pentium M a 1.6Ghz e com 504 Mb de memória.

As 10 instâncias de teste foram retiradas de um conjunto de instâncias referidas em (Trigeiro, Thomas et al. 1989) e foram geradas aleatoriamente com base em cinco características: dimensão do problema, variabilidade da procura, capacidade disponível,

periodicidade de encomenda ("time between orders") e tempo de preparação. Na mesma referência podem encontrar-se informações mais pormenorizadas acerca destas características e da forma como as instâncias foram construídas.

As instâncias que foram seleccionadas têm 20 períodos e podem ser divididas em dois grupos, uma vez que das primeiras 5 instâncias para as restantes 5, o único parâmetro que varia é o número de artigos, que passa de 10 para 20, respectivamente. Pretende-se assim avaliar em termos computacionais o efeito da duplicação do número de artigos.

Numa primeira fase, resolveu-se para as 10 instâncias de teste a relaxação linear da formulação clássica, através de um "solver" de problemas de programação inteira mista, o CPLEX e através do ADDing resolveu-se a relaxação linear da decomposição por artigo e a relaxação linear da decomposição por período, com o objectivo de avaliar a qualidade dos limites inferiores e de avaliar os tempos de resolução. Para avaliar estas medidas de desempenho estabeleceram-se comparações entre a decomposição por artigo e a decomposição por período e entre estes dois modelos de decomposição e o CPLEX. Na Tabela 11 apresentam-se os tempos de resolução da relaxação linear das instâncias de teste, obtidos com o CPLEX, com a decomposição por artigo e com a decomposição por período e na Tabela 12 os valores da solução óptima da relaxação linear (limites inferiores).

Instância	N.º Períodos	N.º Artigos	Decomposição por Artigo Tempo (segundos)	Decomposição por Período Tempo (segundos)	CPLEX Tempo (segundos)
X11219A	20	10	3.52	1.89	0.37
X11229A	20	10	3.03	1.42	0.06
X11418A	20	10	2.95	1.37	0.05
X12219A	20	10	2.72	0.78	0.04
X12429A	20	10	2.78	0.96	0.02
X31219A	20	30	6.61	*	0.06
X31229A	20	30	6.28	*	0.07
X31418A	20	30	7.62	*	0.06
X32219A	20	30	4.83	635.34	0.05
X32429A	20	30	6.67	470.72	0.05

Tabela 11: Tempos de resolução da relaxação linear do conjunto das 10 instâncias.
* Tempo limite (3600 segundos) atingido.

Instância	N.º Períodos	N.º Artigos	Decomposição por Artigo Limite Inferior	Decomposição por Período Limite Inferior	CPLEX Limite Inferior
X11219A	20	10	42244.30	25733.90	25033.48
X11229A	20	10	43023.30	24873.10	24503.86
X11418A	20	10	89626.05	52527.77	50508.18
X12219A	20	10	48010.27	34477.18	33908.46
X12429A	20	10	74501.30	51229.66	49415.84
X31219A	20	30	137612.26	*	79494.33
X31229A	20	30	139518.87	*	76414.12
X31418A	20	30	235567.71	*	127409.01
X32219A	20	30	122513.03	74897.31	74770.47
X32429A	20	30	17125297.76	129860.80	129244.32

Tabela 12: Valor das soluções óptimas da relaxação linear do conjunto das 10 instâncias.
* Quando o tempo limite (3600 segundos) foi atingido ainda não tinha sido encontrado o limite inferior.

Analisando os tempos de resolução da relaxação linear da decomposição por artigo, verifica-se que para resolver o conjunto de instâncias com 30 artigos, demora-se aproximadamente o dobro do tempo, que se demora a resolver o conjunto de instâncias com 10 artigos, que à excepção desta característica são semelhantes.

Quanto à decomposição por período, verifica-se que, para instâncias grandes, este modelo de decomposição é pouco eficiente pois tem tempos de resolução muito elevados quando comparado com os outros dois modelos. No entanto, para as instâncias mais pequenas, esta decomposição é mais eficiente que a decomposição por artigo.

Pode dizer-se que o CPLEX para o conjunto de instâncias resolvidas, é muito eficiente em termos dos tempos de resolução da relaxação linear, quando comparado com os dois modelos de decomposição.

Analisando a Tabela 12, verifica-se que os limites inferiores da decomposição por artigo, são para o conjunto das 10 instâncias muito melhores que os produzidos quer pelo CPLEX, quer pela decomposição por período. Estabelecendo o paralelo entre a decomposição por período e o CPLEX em termos da qualidade dos seus limites inferiores, pode dizer-se que os limites inferiores da decomposição por período são ligeiramente melhores que os do CPLEX.

Após se ter resolvido a relaxação linear das 10 instâncias, resolveu-se numa segunda fase o problema inteiro, através do CPLEX, da decomposição por artigo e da decomposição por período, tendo sido utilizadas as duas estratégias de pesquisa da árvore referidas na secção 5.1. Como medidas de desempenho principais consideraram-se os tempos de resolução, o valor da solução incumbente e a qualidade dos intervalos de integralidade.

Na primeira estratégia utilizou-se a pesquisa primeiro em profundidade, deu-se prioridade ao nodo de pesquisa que tem associada a restrição de partição do tipo ≥ e na regra de partição escolheu-se a variável com parte fraccionária mais próxima de *1*.

Nenhuma das instâncias foi resolvida em menos de 3600 segundos utilizando os modelos de decomposição e a primeira estratégia de pesquisa. Utilizando o CPLEX também nenhuma das instâncias foi resolvida em menos de 3600 segundos, o que nos leva a considerar que quando estamos perante instâncias de uma dimensão considerável, o LPMACC é um problema muito difícil de resolver em termos exactos. Refere-se que do conjunto de instâncias referidas em (Trigeiro, Thomas et al. 1989) as instâncias com 30 artigos são as de maior dimensão.

Na Tabela 13 apresenta-se para a primeira estratégia de pesquisa da árvore o valor das soluções incumbentes do conjunto das 10 instâncias e na Tabela 14 apresentam-se os respectivos intervalos de integralidade. Os intervalos de integralidade dos modelos de decomposição e do CPLEX foram calculados através da seguinte expressão: $\frac{Z_{melhor\ incumbente} - Z_{limite\ inferior}}{Z_{melhor\ incumbente}}$, representando $Z_{melhor\ incumbente}$ o valor da melhor solução inteira encontrada até se ter atingido os 3600 segundos e $Z_{limite\ inferior}$, o valor do limite inferior.

Instância	N.º Períodos	N.º Artigos	Decomposição por Artigo Solução Incumbente	Decomposição por Período Solução Incumbente	CPLEX Solução Incumbente
X11219A	20	10	43521.8	46984.4	43310.0
X11229A	20	10	43765.5	47729.9	44122.6
X11418A	20	10	91036.3	133123.3	93861.9
X12219A	20	10	49458.2	55083.2	49524.7
X12429A	20	10	79208.9	91595.9	81018.1
X31219A	20	30	144653.4	*	139280.4
X31229A	20	30	*	*	*
X31418A	20	30	238250.1	*	244391.1
X32219A	20	30	127220.3	*	123564.8
X32429A	20	30	*	*	*

Tabela 13: Valor das soluções incumbentes do conjunto das 10 instâncias – 1ª estratégia.
* Não foi encontrada nenhuma solução incumbente até o tempo limite (3600 segundos) ter sido atingido.

Instância	N.º Períodos	N.º Artigos	Valor da Melhor Solução Incumbente	Gap - Decomposição por Artigo (%)	gap - Decomposição por Período (%)	gap - CPLEX (%)
X11219A	20	10	43310	2.46	40.58	42.20
X11229A	20	10	43765.5	1.70	43.17	44.01
X11418A	20	10	91036.3	1.55	42.30	44.52
X12219A	20	10	49458.2	2.93	30.29	31.44
X12429A	20	10	79208.9	5.94	35.32	37.61
X31219A	20	30	139280.4	1.20	*	42.92
X31418A	20	30	238250.1	1.13	*	46.52
X32219A	20	30	123564.8	0.85	39.39	39.49

Tabela 14: Intervalos de integralidade (ou gaps) do conjunto das 10 instâncias, atendendo à melhor solução incumbente – 1ª estratégia.
* Não foi calculado o gap por falta do valor do limite inferior.

As soluções incumbentes que se apresentam na Tabela 13 dizem respeito à melhor solução inteira encontrada até se ter atingido o tempo limite de resolução. Utilizando a primeira estratégia, na decomposição por período para todas as instâncias com 10 artigos é encontrada uma solução incumbente até se ter atingido o tempo limite de resolução, mas para as instâncias com 30 artigos não. As soluções incumbentes quer da decomposição por artigo, quer do CPLEX, são melhores que as da decomposição por período e em 5 das instâncias as soluções incumbentes da decomposição por artigo são de melhor qualidade que as do CPLEX.

É de salientar que o software CPLEX é um software comercial de estado-da-arte, o que se reflecte, por exemplo, no número de nodos pesquisados. Analisando a título exemplificativo a instância X11418A, verifica-se que o número de nodos pesquisados no modelo da decomposição por artigo, até se ter atingido os 3600 segundos, foi de 6 054 , ao passo que no mesmo tempo com o CPLEX são pesquisados 765 354 nodos. A vantagem da abordagem de decomposição é contudo clara: mesmo com um número de nodos pesquisados muito inferior, a solução incumbente da decomposição por artigo é melhor que a do CPLEX.

Na coluna 4 da Tabela 14, relativa ao valor da melhor solução incumbente, inscreve-se em cada célula o valor da solução incumbente da decomposição por artigo ou da decomposição por período, ou o valor da solução incumbente do CPLEX, consoante aquele que apresente o valor mais baixo (mais próximo do óptimo inteiro). As instância X31229A e X32429A não se consideram nesta análise, pois para estas instâncias não se encontraram soluções incumbentes com nenhum dos métodos de resolução utilizados.

Utilizando a primeira estratégia de pesquisa o gap médio da decomposição por artigo, da decomposição por período e do CPLEX ronda os 2%, 39% e 41%, respectivamente. O gap da decomposição por período é extremamente elevado, quando comparado com o gap da decomposição por artigo. Comparando o gap da decomposição por período com o do CPLEX,

concluí-se que são ambos semelhantes, embora o gap da decomposição por período seja um pouco melhor.

Na segunda estratégia, para resolver o problema inteiro utilizou-se a pesquisa primeiro em profundidade, deu-se prioridade ao nodo de pesquisa que tem associada a restrição de partição do tipo ≤ e na regra de partição escolheu-se a variável com parte fraccionária mais próxima de *0*. As medidas de desempenho aqui consideradas foram também os tempos de resolução, o valor da solução incumbente e a qualidade dos intervalos de integralidade.

Utilizando a segunda estratégia de pesquisa, tal como na primeira estratégia, nenhuma das instâncias foi resolvida em menos de 3600 segundos. Utilizando o CPLEX também nenhuma das instâncias foi resolvida em menos de 3600 segundos.

Na Tabela 15 apresenta-se para a segunda estratégia de pesquisa da árvore o valor das soluções incumbentes do conjunto das 10 instâncias e na Tabela 16 apresentam-se os respectivos intervalos de integralidade. Tal como na estratégia anterior, para calcular os intervalos de integralidade da decomposição por artigo e compará-los com os do CPLEX, seguiu-se o mesmo procedimento que foi utilizado na primeira estratégia de pesquisa.

Instância	N.º Períodos	N.º Artigos	Decomposição por Artigo Solução Incumbente	Decomposição por Período Solução Incumbente	CPLEX Solução Incumbente
X11219A	20	10	46098.3	*	43310.0
X11229A	20	10	45822.0	*	44122.6
X11418A	20	10	93798.4	96738.8	93861.9
X12219A	20	10	53079.5	*	49524.7
X12429A	20	10	*	*	81018.1
X31219A	20	30	*	*	139280.4
X31229A	20	30	*	*	*
X31418A	20	30	239241.2	*	244391.1
X32219A	20	30	126814.3	*	123564.8
X32429A	20	30	*	*	*

Tabela 15: Valor das soluções incumbentes do conjunto das 10 instâncias – 2ª estratégia.
* Não foi encontrada nenhuma solução incumbente até o tempo limite (3600 segundos) ter sido atingido.

Instância	N.º Períodos	N.º Artigos	Valor da Melhor Solução Incumbente	gap - Decomposição por Artigo (%)	gap -CPLEX (%)
X11219A	20	10	43310	2.46	42.20
X11229A	20	10	44122.6	2.49	44.46
X11418A	20	10	93798.4	4.45	46.15
X12219A	20	10	49524.7	3.06	31.53
X12429A	20	10	81018.1	8.04	39.01
X31219A	20	30	139280.4	1.20	42.92
X31418A	20	30	239241.2	1.54	46.74
X32219A	20	30	123564.8	0.85	39.49

Tabela 16: Intervalos de integralidade (ou gaps) do conjunto das 10 instâncias, atendendo à melhor solução incumbente – 2ª estratégia.

As soluções incumbentes que se apresentam na Tabela 15, dizem respeito à melhor solução inteira encontrada até se ter atingido o tempo limite de resolução. Na decomposição por período, apenas numa das instâncias foi encontrada uma solução incumbente até se ter atingido o tempo limite de resolução, pelo que nesta estratégia não se calcularam os intervalos de integralidade. Para a maioria das instâncias, as soluções incumbentes do CPLEX são de melhor qualidade que as soluções incumbentes da decomposição por artigo. Contudo, para as instâncias X11418A e X31418 o valor da solução incumbente da decomposição por artigo é melhor. Embora fosse necessário realizar mais testes e analisar mais em detalhe as características de cada uma das instâncias escolhidas, uma característica que parece diferenciar estas duas instâncias é o facto de nestas termos capacidades menos apertadas que nas restantes instâncias.

Na coluna 4 da Tabela 16, relativa ao valor da melhor solução incumbente, inscreve-se em cada célula o valor da solução incumbente da decomposição por artigo, ou o valor da solução incumbente do CPLEX, consoante aquele que apresente o valor mais baixo (mais próximo do óptimo inteiro). As instância X31229A e X32429A não se consideram nesta análise, pois para estas instâncias não se encontraram soluções incumbentes com nenhum dos métodos de resolução utilizados.

O gap do CPLEX é extremamente elevado, quando comparado com o gap da decomposição por artigo no conjunto das 8 instâncias de teste. O gap de integralidade da decomposição por artigo ronda em termos médios os 3% e o do CPLEX ronda os 42%.

Em termos gerais para as instâncias testadas os limites inferiores da decomposição por período são de melhor qualidade que os do CPLEX, mas pouco melhores, concluindo-se que a diferença não é significativa em relação à formulação clássica. Para as instâncias com 10 artigos com a decomposição por período demora-se menos tempo a resolver a relaxação linear das instâncias que com a decomposição por artigo, mas o limite inferior da decomposição por artigo

é claramente melhor. Para as instâncias maiores, na decomposição por período o tempo de resolução da relaxação linear é substancialmente mais elevado que o tempo de resolução com a decomposição por artigo.

Na resolução do problema inteiro para as instâncias maiores na decomposição por período não são encontradas soluções incumbentes. Os intervalos de integralidade da decomposição por artigo, são muito baixos quando comparados com os obtidos com a decomposição por período e com o CPLEX, constituindo um motivo para o investimento na melhoria deste método.

Na decomposição por artigo analisando apenas as 5 instâncias que têm 10 artigos na primeira estratégia de pesquisa da árvore, verifica-se que o tempo gasto a resolver os PMR nos nodos da árvore de pesquisa é em termos médios duplo do tempo gasto a resolver os subproblemas nos nodos da árvore. Já na decomposição por período, fazendo a mesma análise para o mesmo grupo de instâncias e para a mesma estratégia de pesquisa da árvore, verifica-se que neste caso o tempo gasto a resolver os PMR nos nodos da árvore de pesquisa é em termos médios quatro vezes superior ao tempo gasto a resolver os subproblemas nos nodos da árvore.

Quer na decomposição por período, quer na decomposição por artigo, na estratégia de pesquisa da árvore, em que se utiliza a pesquisa primeiro em profundidade, se dá prioridade ao nodo de pesquisa que tem associada a restrição de partição do tipo $\geq$ e na regra de partição se utiliza uma estratégia baseada na escolha da variável com parte fraccionária mais próxima de *1*, os resultados são melhores em termos do valor das soluções incumbentes até se atingir o tempo limite de resolução, concluindo-se que esta estratégia conduz a uma melhor exploração da árvore de pesquisa.

6 Conclusões Gerais e Trabalho Futuro

Neste trabalho apresentaram-se dois modelos de decomposição para resolver o problema de lotes de produção multi-artigo capacitado com tempos de preparação e aplicou-se a ambos os modelos, o método de partição e geração de colunas para determinar a solução óptima inteira do problema.

Em ambos os modelos de decomposição aplicou-se a decomposição de Dantzig-Wolfe à formulação de programação inteira mista do problema LPMACC e resolveu-se o problema reformulado através do método de partição e geração de colunas.

No primeiro modelo de decomposição que se denominou por decomposição por artigo, as restrições de capacidade da formulação clássica do LPMACC representam as restrições de ligação e no segundo modelo de decomposição que se denominou por decomposição por período são as restrições de equilíbrio de stocks que definem as restrições de ligação.

Neste trabalho mostrou-se que a decomposição de um problema de programação inteira mista, neste caso a decomposição do problema de lotes de produção multi-artigo capacitado com tempos de preparação, permite obter reformulações cuja relaxação linear produz melhores limites inferiores, fazendo com que, porventura, se possam resolver problemas de maior dimensão até à optimalidade.

Foram estudados para o mesmo problema dois modelos de decomposição diferentes, com o objectivo de estabelecer comparações entre ambos, e de mostrar que quando se aplicam modelos de decomposição diferentes ao mesmo problema, o desempenho desses modelos pode ser bastante diferente. Para as instâncias que se estudaram nos testes computacionais é notória a diferença em termos de desempenho dos dois modelos de decomposição. Nos testes computacionais estes dois modelos de decomposição foram também comparados com um "package" comercial de programação inteira mista.

Para cada modelo de decomposição analisaram-se diferentes alternativas para a obtenção de um primeiro problema mestre restrito válido.

Para resolver o problema inteiro propôs-se um esquema de partição baseado nas variáveis originais, no qual se adicionam restrições de partição ao problema mestre, garantindo-se deste modo a preservação da estrutura do subproblema. Este esquema de partição tem a vantagem de poder ser aplicado a qualquer tipo de problema de programação inteira ou de programação inteira mista.

Na resolução do problema inteiro, testaram-se duas estratégias de pesquisa da árvore.

Os testes computacionais realizados mostraram que o problema de lotes de produção multi-artigo capacitado com tempos de preparação é um problema muito difícil de resolver. Ambos os modelos de decomposição dão melhores limites inferiores que os limites inferiores que se obtêm ao resolver a relaxação linear da formulação clássica. Contudo os limites inferiores da decomposição por artigo são bastante melhores que os da decomposição por período.

Em geral para as instâncias testadas os resultados obtidos com a decomposição por artigo comparativamente com os resultados obtidos com o CPLEX foram melhores, nomeadamente em termos da qualidade das soluções quando é atingido o tempo limite de resolução.

Pode dizer-se que a decomposição por período não é computacionalmente eficiente em termos da obtenção de soluções admissíveis, para o conjunto das instâncias testadas.

Com os testes computacionais que se realizaram não se conseguem melhorias nos tempos de resolução das instâncias comparando os modelos de decomposição com o CPLEX, quer na resolução da relaxação linear, quer na resolução do problema inteiro. Contudo, os tempos de resolução dos modelos de decomposição podem ser melhorados e consequentemente a performance do algoritmo de partição e geração de colunas, caso se implementem algumas melhorias nesse algoritmo e se utilizem algumas das potencialidades do ADDing que não foram nesta fase testadas.

No âmbito do algoritmo de partição e geração de colunas, as principais melhorias a desenvolver num prosseguimento deste trabalho estão relacionadas com:

- a implementação de um algoritmo de resolução eficiente dos subproblemas da decomposição por artigo, neste caso o algoritmo de Wagner-Whitin (Wagner and Whitin 1958);
- a implementação de heurísticas para a obtenção de soluções admissíveis de melhor qualidade, quer no PMR da raiz da árvore de pesquisa, quer nos nodos da árvore de ambas as decomposições;
- realização de testes computacionais mais exaustivos, onde entre outras coisas se efectue a comparação da regra de partição usada neste trabalho com outras regras de partição e se efectue uma selecção mais criteriosa das variáveis de partição;
- a integração dos dois modelos de decomposição, através da aplicação da decomposição de Dantzig-Wolfe múltipla (Alvelos 2005).

Tal como referido anteriormente existem algumas opções incorporadas no ADDing que não foram testadas, que poderão influenciar a performance do algoritmo de partição e geração de colunas. Entre elas destacam-se:

- a remoção de colunas do PMR;
- a gestão dinâmica das restrições de convexidade. Usando esta opção, as restrições de convexidade não estão presentes no PMR e só se inserem no PMR as restrições que numa dada iteração do algoritmo de geração de colunas são violadas;
- a gestão dinâmica das restrições de ligação, que funciona do mesmo modo que a gestão dinâmica das restrições de convexidade.

Referências

Alvelos, F. (2005), *Branch-and-price and multicommodity flows*, Tese de Doutoramento, Escola de Engenharia, Universidade do Minho.

Armentano, V., França, P. and Toledo, F. (1999). *A network flow model for the capacitated lot-sizing problem*. Omega 27: 275-284.

Bahl, H. (1983). *Column generation based heuristic algorithm for multi-item scheduling*. AIIE Transactions 15(2): 136-141.

Barany, I., Roy, T. V. and Wolsey, L. A. (1984). *Strong formulations for multi-item capacitated lot sizing*. Management Science 30(10): 1255-1261.

Barnhart, C., Johnson, E., Nemhauser, G., Savelsbergh, M. and Vance, P. (1998). *Branch-and-price: column generation for solving huge integer programs*. Operations Research 46(3): 316-329.

Belvaux, G. and Wolsey, L. A. (2000). *bc-prod: A specialized branch-and-cut system for lot-sizing problems*. Management Science 46(5): 724-738.

Bitran, G. and Yanasse, H. (1982). *Computational complexity of the capacitated lot size problem*. Management Science 28(10): 1174-1186.

Carvalho, J. M. V. d. (2000). *Using extra dual cuts to accelerate column generation*. to appear in INFORMS Journal on Computing.

Carvalho, J. M. V. d. (2002). *LP models for bin packing and cutting stock problems*. European Journal of Operational Research 141: 253-273.

Cattrysse, D., Maes, J. and Wassenhove, L. (1990). *Set partitioning and column generation heuristic for capacitated dynamic lotsizing*. European Journal of Operational Research 46: 38-47.

Chen, W. H. and Thizy, J. M. (1990). *Analysis of relaxations for the multi-item capacitated lot-sizing problem*. Annals of Operations Research 26: 29-72.

Constantino, M. (1996). *A cutting plane approach to capacitated lot-sizing with start-up costs*. Mathematical Programming 75: 353-376.

Dantzig, G. B. and Wolfe, P. (1960). *Decomposition principle for linear programs*. Operations Research 8: 101-111.

Desrosiers, J., Dumas, Y., Solomon, M. M. and Soumis, F. (1995). Time constrained routing and scheduling. Network Routing. M. O. B. e. al. Amsterdam, Elsevier Science**:** 35-139.

Diaby, M., Bahl, H. C., Karwan, M. H. and Zionts, S. (1992a). *Capacitated lot-sizing and scheduling by lagrangean relaxation*. European Journal of Operational Research 59: 444-458.

Diaby, M., Bahl, H. C., Karwan, M. H. and Zionts, S. (1992b). *A lagrangean relaxation approach for very-large-scale capacitated lot-sizing*. Management Science 38(9): 1329-1340.

Dixon, P. and Silver, E. A. (1981). *A heuristic solution procedure for the multi-item, single-level, limited capacity, lot-sizing problem*. Journal of Operations Management 2(1): 23-39.

Dogramaci, A., Panayiotopoulos, J. C. and Adam, N. R. (1981). *The dynamic lot-sizing problem for multiple items under limited capacity*. AIIE Transactions 13(4): 294-303.

Dzielinski, B. P. and Gomory, R. E. (1965). *Optimal programming of lot sizes, inventory and labor allocations*. Management Science, Series A 11(9): 874-890.

Eisenhut, P. S. (1975). *A dynamic lot sizing algorithm with capacity constraints*. AIIE Transactions 7(2): 170-176.

Eppen, G. D. and Martin, R. K. (1987). *Solving multi-item capacitated lot-sizing problems using variable redefinition*. Operations Research 35(6): 832-848.

Fleischmann, B. (1990). *The discrete lot-sizing and scheduling problem*. European Journal of Operational Research 44(3): 337-348.

Florian, M. and Klein, M. (1971). *Deterministic production planning with concave costs and capacity constraints*. Management Science 18(1): 12-20.

Florian, M., Lenstra, J. K. and Kan, A. H. G. R. (1980). *Deterministic production planning algorithms and complexity*. Management Science 26(7): 669-679.

Geoffrion, A. M. (1974). *Lagrangean relaxation for integer programming*. Mathematical Programming Study 2: 82-114.

Harris, F. W. (1913). *How many parts to make at once*. Factory, The Magazine of Management 10(2).

Hoesel, S. V. and Kolen, A. (1994). *A linear description of the discrete lot-sizing and scheduling problem*. European Journal of Operational Research 72(2): 342- 364.

ILOG (2002). *CPLEX 8.0, User's Manual*.

Jans, R. (2002), *Capacitated lot sizing problems: new applications, formulations and algorithms*, PhD Thesis, Katholieke Universiteit.

Karimi, B., Ghomia, S. M. T. F. and Wilson, J. M. (2003). *The capacitated lot sizing problem: a review of models and algorithms*. Omega 31: 365-378.

Karmarkar, U. S., Kekre, S. and Kekre, S. (1987). *The deterministic lot sizing problem with startup and reservation costs*. Operations Research 35(3): 389-398.

Karmarkar, U. S. and Schrage, L. (1985). *The deterministic dynamic product cycling problem*. Operations Research 33(2): 326-345.

Kirka, O. and Kokten, M. (1994). *A new heuristic approach for the multi-item dynamic lot sizing problem*. European Journal of Operational Research 75(2): 332-341.

Kuik, R., Salomon, M. and Wassenhove, L. (1994). *Batching decisions: structure and models*. European Journal of Operational Research 75(Lotsizing models for production planning): 243-263.

Lambrecht, M. R. and Vanderveken, H. (1979). *Heuristic procedures for the single operation, multi-item loading problem*. AIIE Transactions 11(4): 319-326.

Lasdon, L. S. and Terjung, R. C. (1971). *An efficient algorithm for multi-item scheduling*. Operations Research 19(4): 946-969.

Lübbecke, M. E. and Desrosiers, J. (2002). Selected topics in column generation, GERARD,

Les Cahiers de GERAD G-2002-64.

Maes, J., McClain, J. O. and Wassenhove, L. N. (1991). *Multilevel capacitated lotsizing complexity and LP-based heuristics*. European Journal of Operational Research 53(2): 131-148.

Maes, J. and Wassenhove, L. N. V. (1986a). *Multi item single level capacitated dynamic lotsizing heuristics: a computational comparison (part I: static case)*. IIE Transactions: 114-123.

Maes, J. and Wassenhove, L. N. V. (1986b). *Multi item single level capacitated dynamic lotsizing heuristics: a computational comparison (part II: rolling horizon)*. IIE Transactions: 124-129.

Manne, A. S. (1958). *Programming of economic lot sizes*. Management Science 4(2): 115-135.

Martin, R. K. (1987). *Generating alternative mixed-integer programming models using variable redefinition*. Operations Research 35(6): 820-831.

Merle, O. d., Goffin, J. L., Trouiller, C. and Vial, J. (1999). A lagrangian relaxation of the capacitated multi-item lot sizing problem solved with an interior point cutting plane algorithm. Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems. P. M. Pardalos, Kluwer Academic Publishers.

Miller, A. J., Nemhauser, G. L. and Savelsbergh, M. W. P. (2000). Solving multi-Item capacitated lot-sizing problems with setup times by branch-and-cut, CORE discussion paper 2000/39.

Miller, A. J. and Wolsey, L. A. (2001). Discrete lot-sizing and convex integer programming, CORE Discussion Paper 2001.

Nemhauser, G. L. and Wolsey, L. A. (1999). *Integer and Combinatorial Optimization*, John Wiley and Sons.

Pochet, Y. and Wolsey, L. A. (1991). *Solving multi-item lot-sizing problems using strong cutting planes*. Management Science 37(1): 53-67.

Rizk, N. and Martel, A. (2001). *Supply chain flow planning methods - a review of the lot-sizing literature*. (disponível a 9 de Maio de 2005 em http://www.forac.ulaval.ca/Centor/Documents/Documents%20de%20travail%202001/DT-2001-AM-1.pdf).

Salomon, M. (1991). *Deterministic Lotsizing Models for Production Planning*, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag.

Silver, E. A. and Meal, H. C. (1973). *A heuristic for selecting lot size quantities for the case of a deterministic time varying demand rate and discrete opportunities for replenishment*. Production and Inventory Management 14(2): 64-74.

Thizy, J. M. and Wassenhove, L. N. V. (1985). *Lagrangean relaxation for the multi-item capacitated lot-sizing problem: a heuristic implementation*. IIE Transactions 17(4): 308-313.

Trigeiro, W. W., Thomas, L. J. and McClain, J. O. (1989). *Capacitated lot sizing with setup times*. Management Science 35(3): 353-366.

Vanderbeck, F. (1994), *Decomposition and column generation for integer programs*, PhD Thesis, Faculté des Sciences Appliquées, Université Catholique de Louvain.

Vollmann, T. E., Berry, W. L. and Whybark, D. C. (1997). *Manufacturing Planning and Control Systems*, McGraw-Hill.

Wagner, H. and Whitin, T. (1958). *Dynamic version of the economic lot size model*. Management Science 5(1): 89-96.

Wilhelm, W. E. (2001). *A technical review of column generation in integer programming*. Optimization and Engineering 2(2): 159-200.

Zangwill, W. I. (1969). *A backlogging model and a multi-echelon model of a dynamic economic lot size production system - a network approach*. Management Science 15(9): 506-527.

Anexo – Classes de C++

instance_ls.h

```
#if !defined _INSTANCE_LS
#define _INSTANCE_LS

#include "timer.h"
#include "pars.h"
#include "myheader.h"

namespace fillbpc {

/// Class that defines an instance of the capacitated lot sizing with setup
times problem

/*
Supports the Trigeiro et al. format.
*/

class instance_ls
{
public:

       // Constructor
       // Almost empty
       // Memory allocations are not performed here

       instance_ls(char *aName, char aFormat);

       // Destructor
       // Memory deallocations

       virtual ~instance_ls();

       // Read instance from file named Name and with format
       // Type
       // If the instance cannot be loaded, the execution will
       // be aborted and some message will appear on cerr
       // All memory allocations are performed here

       void ReadFromFile(char *Name, char Type);

       // Writes instance in cout
       // Descriptive format!

       void WriteDescription();

       void WriteFileGeneralInfo(std::ofstream &aFile);

       // Get methods are used to access private data

       int GetNumPers() { return NumPers; };
       int GetNumItems() { return NumItems; };
       int GetCapacity() { return Capacity;};
       int GetProdCost() { return ProdCost;};
       double GetProdTime(int j) { return ProdTime[j];};
       double GetHoldCost(int j) { return HoldCost[j];};
       int GetSetupTime(int j) { return SetupTime[j];};
       int GetSetupCost(int j) { return SetupCost[j];};
       int GetDemand(int k, int j) { return Demand[k][j];};
       double GetArtCost() {return CostArt;};
       double GetMaxProdPerItem(int k, int j) { return MaxProdPerItem[k][j];};

private:

       /// Number of periods.
```

```
	int NumPers;

	/// Number of items.

	int NumItems;

	/// Capacity per period.

	/**
	In the Trigeiro et al. instances, capacity is constant over all periods.
	*/

	int Capacity;

	/// Production cost.

	/**
	In the Trigeiro et al. instances, production cost is constant over all
periods and items.
	*/

	int ProdCost;

	/// Production time.

	/**
	In the Trigeiro et al. instances, production time is constant over all
periods,
	(possibly) being different for different items.
	*/

	double *ProdTime;

	/// Holding cost.

	/**
	In the Trigeiro et al. instances, holding cost is constant over all
periods,
	(possibly) being different for different items.
	*/

	double *HoldCost;

	/// Setup time.

	/**
	In the Trigeiro et al. instances, setup time is constant over all
periods,
	(possibly) being different for different items.
	*/

	int *SetupTime;

	/// Setup cost.

	/**
	In the Trigeiro et al. instances, setup cost is constant over all
periods,
	(possibly) being different for different items.
	*/

	int *SetupCost;

	/// Demand.

	/**
	First index is about the period and the second one about the item.
```

```
	*/

	int **Demand;

	/// Artificial cost

	/**
	To be computed based on the concrete instance.
	*/

	double CostArt;

	/// Data structure for keeping upper bound of the quantity produced in a
given period for a given item.

	/**
	Calculated when reading the instance.
	*/

	double **MaxProdPerItem;
};

} // end of namespace fillbpc

#endif
```

instance_ls.cpp

```
#include "myheader.h"

#include "instance_ls.h"

#include <string.h>

using namespace fillbpc;

extern bool IsNumInteger(double &aNumero, double aEps);

instance_ls::instance_ls(char *aName, char aFormat)
{

	ReadFromFile(aName,aFormat);

}

instance_ls::~instance_ls()
{

	int k;
	delete [] ProdTime;
	delete [] HoldCost;
	delete [] SetupCost;
	delete [] SetupTime;
	for (k=0;k<NumPers;k++) {
		delete [] Demand[k];
		delete [] MaxProdPerItem[k];
	}
	delete [] Demand;
	delete [] MaxProdPerItem;

}

void instance_ls::ReadFromFile(char *Name, char Type)
{

	double CostMax=0, TotalDem=0;
	int i,j,k;
```

```
	/*********************************************************/
	// Trigeiro et al. format
	/*********************************************************/
	if (Type=='t') {

		ifstream inFile (Name,ios::in);
		if (!inFile) {
			cout << endl << "File does not exist" << endl;
			exit(-1);
		}

		double MaxFixedCost=0, MaxVarCost=0, TotalDem=0;

		inFile >> NumItems >> NumPers >> ProdCost >> Capacity;

		MaxVarCost=ProdCost;

		ProdTime = new double [NumItems];
		HoldCost = new double [NumItems];
		SetupCost = new int [NumItems];
		SetupTime = new int [NumItems];

		char trashchar;
		for (j=0;j<NumItems;j++) {
			inFile >> ProdTime[j] >> HoldCost[j] >> SetupTime[j] >>
trashchar;
			if (trashchar!='.') {
				cout << endl << "Error reading the instance: check
that the format of the file is really the Trigeiro et al. one";
				exit(-1);
			}
			inFile >> SetupCost[j] >> trashchar;
			if (trashchar!='.') {
				cout << endl << "Error reading the instance: check
that the format of the file is really the Trigeiro et al. one";
				exit(-1);
			}
			if (HoldCost[j]>MaxVarCost) MaxVarCost=HoldCost[j];
			if (SetupCost[j]>MaxFixedCost) MaxFixedCost=SetupCost[j];
		}

		Demand=new int *[NumPers];
		for (k=0;k<NumPers;k++) {
			Demand[k]=new int [NumItems];
			for (j=0;j<NumItems;j++) {
				inFile >> Demand[k][j];
				TotalDem+=Demand[k][j];
			}
		}

	CostArt=TotalDem*MaxVarCost*NumItems*NumPers+NumPers*MaxFixedCost*NumIte
ms;

		inFile.close();

	}

	MaxProdPerItem=new double *[NumPers];
	for (k=0;k<NumPers;k++) {
		MaxProdPerItem[k]=new double [NumItems];
	}

	double FutDemItem, CapPerItem;

	for (k=0;k<NumPers;k++) {
		for (j=0;j<NumItems;j++) {
			FutDemItem=0;
```

```
                    for (i=k;i<NumPers;i++) FutDemItem+=Demand[i][j];
                    CapPerItem=((Capacity-SetupTime[j])/ProdTime[j]);
                    if (CapPerItem<FutDemItem)
MaxProdPerItem[k][j]=CapPerItem;
                    else MaxProdPerItem[k][j]=FutDemItem;
             }
      }
}

void instance_ls::WriteDescription()
{

      int j,k;

      cout << endl << endl << "Instance";
      cout << endl << "Number of periods " << NumPers;
      cout << endl << "Number of items " << NumItems;
      cout << endl << "Capacity " << Capacity;
      cout << endl << "Production Cost " << ProdCost;
      cout << endl << endl << "Per
ITEM:\tProdTime\tHoldCost\tSetupTime\tSetupCost";
      for (j=0;j<NumItems;j++) cout << endl << "\t" << ProdTime[j] << "\t" <<
HoldCost[j] << "\t" << SetupTime[j] << "\t" << SetupCost[j];
      cout << endl << endl << "Demand per PERIOD ";
      for (k=0;k<NumPers;k++) {
            cout << endl;
            for (j=0;j<NumItems;j++) cout << Demand[k][j] << "\t";
      }
      cout << endl << endl << "MaxProdPerItem per PERIOD ";
      for (k=0;k<NumPers;k++) {
            cout << endl;
            for (j=0;j<NumItems;j++) cout << MaxProdPerItem[k][j] << "\t";
      }
}

void instance_ls::WriteFileGeneralInfo(ofstream &aFile)
{

      aFile << NumPers << "\tNumber of periods" << endl;
      aFile << NumItems << "\tNumber of items" << endl;
}
```

ls_lot_mip.h

```
#if !defined _LS_LOT_MIP
#define _LS_LOT_MIP

#include "instance_ls.h"

#include "decmodel.h"

namespace fillbpc {

class ls_lot_mip
{

public:

      /// Constructor: pass parameters to base class columngeneration.
      ls_lot_mip(decmodel *aDecModel, instance_ls *aInstance);

      // Destructor.
      virtual ~ls_lot_mip();

private:

};
```

```
} // end of namespace fillbpc
#endif
```

ls_lot_mip.cpp

```
//****************** Product Decomposition***********************************

#include "ls_lot_mip.h"
#include "instance_ls.h"

using namespace fillbpc;

ls_lot_mip::ls_lot_mip(decmodel *aDecModel, instance_ls *aInstance)
{

       //j is the index of the periods and i the index of the items.
       int k,j, i, p;

       //Dimensions of the original problem
       aDecModel->SetDimensions(aInstance->GetNumItems(),3*aInstance-
>GetNumPers(),aInstance->GetNumItems(),aInstance->GetNumItems());

       aDecModel->SetTypeVars(2*aInstance->GetNumPers(),aInstance-
>GetNumPers(),0);

       //convexity constraints
       char *SenseConv=new char [aInstance->GetNumItems()];
       int *RhsConv=new int [aInstance->GetNumItems()];
       bool *PresentConv=new bool [aInstance->GetNumItems()];

       for (i=0;i<aInstance->GetNumItems();i++) {
              SenseConv[i]='G';
              RhsConv[i]=1;
              PresentConv[i]=false;
       }
       aDecModel->SetConv(aInstance->GetNumItems(), SenseConv, RhsConv,
PresentConv);

       delete [] SenseConv;
       delete [] RhsConv;
       delete [] PresentConv;

       //original constraints
       char *SenseOrig=new char [aInstance->GetNumPers()];
       int *RhsOrig=new int [aInstance->GetNumPers()];
       bool *PresentOrig=new bool [aInstance->GetNumPers()];

       for (j=0;j<aInstance->GetNumPers();j++) {
              SenseOrig[j]='L';
              RhsOrig[j]=aInstance->GetCapacity();
              PresentOrig[j]=true;
       }
       aDecModel->SetOrig(aInstance->GetNumPers(), SenseOrig, RhsOrig,
PresentOrig);

       delete [] SenseOrig;
       delete [] RhsOrig;
       delete [] PresentOrig;

       // D Matrix
       aDecModel->SetDEqual(false);

       int **DkBeg=new int *[aInstance->GetNumItems()];
       int **DkInd=new int *[aInstance->GetNumItems()];
       double **DkVal=new double *[aInstance->GetNumItems()];
       int *DkNnz=new int [aInstance->GetNumItems()];
```

```
	for (k=0;k<aInstance->GetNumItems();k++) {
		DkBeg[k]=new int [3*aInstance->GetNumPers()];
		DkInd[k]=new int [2*aInstance->GetNumPers()];
		DkVal[k]=new double [2*aInstance->GetNumPers()];

		for (j=0;j<aInstance->GetNumPers();j++) {
			DkBeg[k][j]=0;
			DkInd[k][j]=j;
			DkVal[k][j]=aInstance->GetProdTime(k);
		}
		for (j=aInstance->GetNumPers();j<2*aInstance->GetNumPers();j++) {
			DkBeg[k][j]=j-aInstance->GetNumPers();
			DkInd[k][j]=j-aInstance->GetNumPers();
			DkVal[k][j]=aInstance->GetSetupTime(k);
		}
		p=j;
		for (j=0;j<aInstance->GetNumPers();j++) {
			DkBeg[k][p]=aInstance->GetNumPers()+j;
			p++;
		}

		DkNnz[k]=2*aInstance->GetNumPers();
	}
	aDecModel->SetDk(DkBeg, DkInd, DkVal, DkNnz);

	for (k=0;k<aInstance->GetNumItems();k++) {
		delete [] DkBeg[k];
		delete [] DkInd[k];
		delete [] DkVal[k];
	}
	delete [] DkBeg;
	delete [] DkInd;
	delete [] DkVal;
	delete [] DkNnz;

	//E matrix
	int *EBeg=new int [aInstance->GetNumItems()];
	int *EInd=new int [aInstance->GetNumItems()];
	double *EVal=new double [aInstance->GetNumItems()];

	for (j=0;j<aInstance->GetNumItems();j++) {
		EBeg[j]=j;
		EInd[j]=j;
		EVal[j]=1;
	}

	aDecModel->SetE(EBeg,EInd,EVal,aInstance->GetNumItems());

	delete [] EBeg;
	delete [] EInd;
	delete [] EVal;

	//Set coefficients of the extra variables in the objective function
	double *ObjExtraVars=new double [aInstance->GetNumItems()];

	for (j=0;j<aInstance->GetNumItems();j++) {
		ObjExtraVars[j]=aInstance->GetArtCost();
	}
	aDecModel->SetObjExtraVars(ObjExtraVars);

	delete [] ObjExtraVars;

	//Set coefficients of the variables in the objective function
	double **ObjVars=new double *[aInstance->GetNumItems()];

	for (k=0;k<aInstance->GetNumItems();k++) {
		ObjVars[k]=new double [3*aInstance->GetNumPers()];
```

```
              for (j=0;j<aInstance->GetNumPers();j++) {
                     ObjVars[k][j]=aInstance->GetHoldCost(k);
              }
              for (j=aInstance->GetNumPers();j<2*aInstance->GetNumPers();j++) {
                     ObjVars[k][j]=aInstance->GetProdCost();
              }
              for (j=2*aInstance->GetNumPers();j<3*aInstance->GetNumPers();j++)
{
                     ObjVars[k][j]=aInstance->GetSetupCost(k);
              }

       }
       aDecModel->SetObjVars(ObjVars);

       for (i=0;i<aInstance->GetNumItems();i++) { delete [] ObjVars[i];
       }
       delete [] ObjVars;

       //Artificial Cost
       aDecModel->SetArtCost(aInstance->GetArtCost());

       //set subproblem
       int *NumConstraintsk=new int [aInstance->GetNumItems()];
       char **Sensek=new char *[aInstance->GetNumItems()];
       double **Rhsk=new double *[aInstance->GetNumItems()];
       int **AkBeg=new int *[aInstance->GetNumItems()];
       int **AkInd=new int *[aInstance->GetNumItems()];
       double **AkVal=new double *[aInstance->GetNumItems()];
       int *AkNnz=new int [aInstance->GetNumItems()];

       for (i=0;i<aInstance->GetNumItems();i++) {
              Sensek[i]=new char [2*aInstance->GetNumPers()];
              Rhsk[i]=new double [2*aInstance->GetNumPers()];
              AkBeg[i]=new int [3*aInstance->GetNumPers()];
              AkInd[i]=new int [5*aInstance->GetNumPers()-1];
              AkVal[i]=new double [5*aInstance->GetNumPers()-1];
       }

       for (k=0;k<aInstance->GetNumItems();k++) {
              NumConstraintsk[k]=2*aInstance->GetNumPers();
              AkNnz[k]=5*aInstance->GetNumPers()-1;
              for (j=0;j<aInstance->GetNumPers();j++) {
                     Sensek[k][j]='G';
                     Rhsk[k][j]=aInstance->GetDemand(j,k);
              }
              for (j=aInstance->GetNumPers();j<2*aInstance->GetNumPers();j++) {
                     Sensek[k][j]='L';
                     Rhsk[k][j]=0;
              }
              for (j=0;j<aInstance->GetNumPers();j++){
                     if (j==0){
                            AkBeg[k][j]=0;
                     }
                     else {
                            AkBeg[k][j]=AkBeg[k][j-1]+2;
                     }
              }
              for (j=aInstance->GetNumPers();j<2*aInstance->GetNumPers();j++){
                     if (j==aInstance->GetNumPers()){
                            AkBeg[k][j]=2*aInstance->GetNumPers()-1;
                     }
                     else{
                            AkBeg[k][j]=AkBeg[k][j-1]+2;
                     }
              }
              for (j=2*aInstance->GetNumPers();j<3*aInstance-
>GetNumPers();j++){
                     if (j==2*aInstance->GetNumPers()){
```

```
                        AkBeg[k][j]=4*aInstance->GetNumPers()-1;
                }
                else{
                        AkBeg[k][j]=AkBeg[k][j-1]+1;
                }
        }
        p=0;
        for (j=0;j<aInstance->GetNumPers()-1;j++){
                AkVal[k][p]=-1;
                AkInd[k][p]=j;
                p++;
                AkVal[k][p]=1;
                AkInd[k][p]=j+1;
                p++;
        }
        AkVal[k][p]=-1;
        AkInd[k][p]=aInstance->GetNumPers()-1;
        p++;

        for (j=0;j<aInstance->GetNumPers();j++){
                AkVal[k][p]=1;
                AkInd[k][p]=j;
                p++;
                AkVal[k][p]=1;
                AkInd[k][p]=j+aInstance->GetNumPers();
                p++;
        }
        for (j=0;j<aInstance->GetNumPers();j++){
                AkVal[k][p]=-aInstance->GetMaxProdPerItem(j,k);
                AkInd[k][p]=j+aInstance->GetNumPers();
                p++;
        }

}
aDecModel>SetSubproblem(NumConstraintsk,Sensek,Rhsk,AkBeg,AkInd,AkVal,Ak
Nnz);

for (i=0;i<aInstance->GetNumItems();i++) {
        delete [] Sensek[i];
        delete [] Rhsk[i];
        delete [] AkBeg[i];
        delete [] AkInd[i];
        delete [] AkVal[i];
}
delete [] Sensek;
delete [] Rhsk;
delete [] AkBeg;
delete [] AkInd;
delete [] AkVal;
delete [] NumConstraintsk;
delete [] AkNnz;
}

ls_lot_mip::~ls_lot_mip()
{
}
```

ls_knps_mip.h

```
#if !defined _LS_KNPS_MIP
#define _LS_KNPS_MIP

#include "instance_ls.h"

#include "decmodel.h"

namespace fillbpc {
```

```
class ls_knps_mip
{

public:

	/// Constructor: pass parameters to base class columngeneration.
	ls_knps_mip(decmodel *aDecModel, instance_ls *aInstance);

	// Destructor.
	virtual ~ls_knps_mip();

private:

};

} // end of namespace fillbpc
#endif
```

ls_knps_mip.cpp

```
//************************ Period Decomposition ****************************

#include "ls_knps_mip.h"
#include "instance_ls.h"

using namespace fillbpc;

ls_knps_mip::ls_knps_mip(decmodel *aDecModel, instance_ls *aInstance)
{

	//j is the index of the periods and i the index of the items.
	int k,j, i;

	//Dimensions of the original problem
	aDecModel->SetDimensions(aInstance->GetNumPers(),2*aInstance-
>GetNumItems(),0,aInstance->GetNumItems()+aInstance->GetNumItems()*aInstance-
>GetNumPers());

	aDecModel->SetTypeVars(aInstance->GetNumItems(),aInstance-
>GetNumItems(),0);

	//convexity constraints
	char *SenseConv=new char [aInstance->GetNumPers()];
	int *RhsConv=new int [aInstance->GetNumPers()];
	bool *PresentConv=new bool [aInstance->GetNumPers()];

	for (j=0;j<aInstance->GetNumPers();j++) {
		SenseConv[j]='L';
		RhsConv[j]=1;
		PresentConv[j]=true;
	}
	aDecModel->SetConv(aInstance->GetNumPers(), SenseConv, RhsConv,
PresentConv);

	delete [] SenseConv;
	delete [] RhsConv;
	delete [] PresentConv;

	//original constraints
	char *SenseOrig=new char [aInstance->GetNumItems()*aInstance-
>GetNumPers()];
	int *RhsOrig=new int [aInstance->GetNumItems()*aInstance->GetNumPers()];
	bool *PresentOrig=new bool [aInstance->GetNumItems()*aInstance-
>GetNumPers()];

	k=0;
```

```
	for (j=0;j<aInstance->GetNumPers();j++) {
		for (i=0;i<aInstance->GetNumItems();i++) {
			SenseOrig[k]='G';
			RhsOrig[k]=aInstance->GetDemand(j,i);
			PresentOrig[k]=true;
			k++;
		}
	}
	aDecModel->SetOrig(aInstance->GetNumItems()*aInstance->GetNumPers(),
SenseOrig, RhsOrig, PresentOrig);

	delete [] SenseOrig;
	delete [] RhsOrig;
	delete [] PresentOrig;

	//D matrix
	aDecModel->SetDEqual(true);

	int *DBeg=new int [2*aInstance->GetNumItems()];
	int *DInd=new int [aInstance->GetNumItems()];
	double *DVal=new double [aInstance->GetNumItems()];

	/*int *DBeg=new int [aInstance->GetNumItems()];
	int *DInd=new int [aInstance->GetNumItems()];
	double *DVal=new double [aInstance->GetNumItems()];*/

	for (i=0;i<aInstance->GetNumItems();i++) {
		DVal[i]=1;
		DInd[i]=i;
		DBeg[i]=i;
	}
	for (i=aInstance->GetNumItems();i<2*aInstance->GetNumItems();i++) {
		DBeg[i]=aInstance->GetNumItems();
	}
	aDecModel->SetD(DBeg, DInd, DVal, aInstance->GetNumItems());

	delete [] DBeg;
	delete [] DInd;
	delete [] DVal;


	//E matrix
	int *EBeg=new int [aInstance->GetNumItems()+aInstance-
>GetNumItems()*aInstance->GetNumPers()];
	int *EInd=new int [aInstance->GetNumItems()+(2*aInstance-
>GetNumItems()*(aInstance->GetNumPers()-1))+aInstance->GetNumItems()];
	double *EVal=new double [aInstance->GetNumItems()+(2*aInstance-
>GetNumItems()*(aInstance->GetNumPers()-1))+aInstance->GetNumItems()];

	k=0;
	for (i=0;i<aInstance->GetNumItems();i++) {
		EBeg[i]=i;
		EInd[i]=i+aInstance->GetNumPers();
		EVal[i]=1;
		k++;
	}

	int ENnz=k;
	for (j=0;j<aInstance->GetNumItems()*(aInstance->GetNumPers()-1);j++) {
		EInd[ENnz]=aInstance->GetNumPers()+j;
		EVal[ENnz]=-1;
		EBeg[k]=ENnz;
		ENnz++;
		EInd[ENnz]=aInstance->GetNumPers()+aInstance->GetNumItems()+j;
		EVal[ENnz]=1;
		ENnz++;
		k++;
	}
```

```
	for (i=0;i<aInstance->GetNumItems();i++) {
		EInd[ENnz]=aInstance->GetNumPers()+aInstance-
>GetNumItems()*(aInstance->GetNumPers()-1)+i;
		EVal[ENnz]=-1;
		EBeg[k]=ENnz;
		ENnz++;
		k++;
	}
	aDecModel->SetE(EBeg,EInd,EVal,ENnz);

	delete [] EBeg;
	delete [] EInd;
	delete [] EVal;

	//Set coefficients of the extra variables in the objective function
	double *ObjExtraVars=new double [aInstance->GetNumItems()+aInstance-
>GetNumItems()*aInstance->GetNumPers()];

	k=0;
	for (i=0;i<aInstance->GetNumItems();i++) {
		ObjExtraVars[k]=aInstance->GetArtCost();
		k++;
	}
	for (j=0;j<aInstance->GetNumPers();j++) {
		for (i=0;i<aInstance->GetNumItems();i++) {
			ObjExtraVars[k]=aInstance->GetHoldCost(i);
			k++;
		}
	}
	aDecModel->SetObjExtraVars(ObjExtraVars);

	delete [] ObjExtraVars;

	//Set coefficients of the variables in the objective function
	double **ObjVars=new double *[aInstance->GetNumPers()];

	for (j=0;j<aInstance->GetNumPers();j++) {
		k=0;
		ObjVars[j]=new double [2*aInstance->GetNumItems()];
		for (i=0;i<aInstance->GetNumItems();i++) {
			ObjVars[j][k]=aInstance->GetProdCost();
			k++;
		}
		for (i=0;i<aInstance->GetNumItems();i++) {
			ObjVars[j][k]=aInstance->GetSetupCost(i);
			k++;
		}
	}
	aDecModel->SetObjVars(ObjVars);

	for (j=0;j<aInstance->GetNumPers();j++) { delete [] ObjVars[j];
	}
	delete [] ObjVars;

	//Artificial Cost
	aDecModel->SetArtCost(aInstance->GetArtCost());

	//set subproblem
	int *NumConstraintsk=new int [aInstance->GetNumPers()];
	char **Sensek=new char *[aInstance->GetNumPers()];
	double **Rhsk=new double *[aInstance->GetNumPers()];
	int **AkBeg=new int *[aInstance->GetNumPers()];
	int **AkInd=new int *[aInstance->GetNumPers()];
	double **AkVal=new double *[aInstance->GetNumPers()];
	int *AkNnz=new int [aInstance->GetNumPers()];

	for (j=0;j<aInstance->GetNumPers();j++) {
		Sensek[j]=new char [aInstance->GetNumItems()+1];
```

```
              Rhsk[j]=new double [aInstance->GetNumItems()+1];
              AkBeg[j]=new int [aInstance->GetNumItems()*2];
              AkInd[j]=new int [aInstance->GetNumItems()*4];
              AkVal[j]=new double [aInstance->GetNumItems()*4];
       }

       for (j=0;j<aInstance->GetNumPers();j++) {
              NumConstraintsk[j]=aInstance->GetNumItems()+1;
              AkNnz[j]=aInstance->GetNumItems()*4;
              for (i=0;i<aInstance->GetNumItems()+1;i++) {
                     Sensek[j][i]='L';
              }
              Rhsk[j][0]=aInstance->GetCapacity();
              for (i=1;i<aInstance->GetNumItems()+1;i++) {
                     Rhsk[j][i]=0;
              }
              for (i=0;i<aInstance->GetNumItems()*2;i++){
                     AkBeg[j][i]=i*2;
              }
              k=0;
              for (i=0;i<aInstance->GetNumItems();i++){
                     AkVal[j][k]=aInstance->GetProdTime(i);
                     AkInd[j][k]=0;
                     k++;
                     AkVal[j][k]=1;
                     AkInd[j][k]=i+1;
                     k++;
              }
              for (i=0;i<aInstance->GetNumItems();i++){
                     AkVal[j][k]=aInstance->GetSetupTime(i);
                     AkInd[j][k]=0;
                     k++;
                     AkVal[j][k]=-aInstance->GetMaxProdPerItem(j,i);
                     AkInd[j][k]=i+1;
                     k++;
              }
       }
       aDecModel>SetSubproblem(NumConstraintsk,Sensek,Rhsk,AkBeg,AkInd,AkVal,Ak
Nnz);

       for (j=0;j<aInstance->GetNumPers();j++) {
              delete [] Sensek[j];
              delete [] Rhsk[j];
              delete [] AkBeg[j];
              delete [] AkInd[j];
              delete [] AkVal[j];
       }
       delete [] Sensek;
       delete [] Rhsk;
       delete [] AkBeg;
       delete [] AkInd;
       delete [] AkVal;
       delete [] NumConstraintsk;
       delete [] AkNnz;
}

ls_knps_mip::~ls_knps_mip()
{
}
```