Uma abordagem híbrida de algoritmo de abelhas e *fix-and-optimize* para o problema de dimensionamento de lotes multiestágio com limitação de capacidade e preservação da preparação

Marcos Mansano Furlan
Maristela Oliveira dos Santos
Universidade de São Paulo-USP
Av. Trabalhador São-carlense, 400 - Centro
Caixa Postal: 668 - CEP: 13560-970 - São Carlos - SP
e-mail: {mafurlan, mari}@icmc.usp.br

Resumo

Este trabalho considera o problema de dimensionamento de lotes em um sistema de produção multiestágio. O problema consiste em determinar um plano de produção que atenda a demanda dos produtos finais e de seus componentes em cada período de um horizonte finito de planejamento, sem atrasos, minimizando os custos envolvidos. Neste trabalho, consideraremos o problema de dimensionamento de lotes multiestágio com limitação de capacidade, preservação da preparação, possibilidade de utilização de hora extra, custos e tempos de preparações positivos e *lead-time* também positivo. Para resolver este problema, propomos uma heurística híbrida de um método baseado na formulação matemática do problema (*fix-and-optimize*) com a metaheurística algoritmo de abelhas. Foram feitos testes com algumas instâncias da literatura e os resultados são comparados com uma outra heurística que aborda o mesmo problema. Os resultados obtidos para as classes de testes mostraram-se promissores.

PALAVRAS CHAVE: **Dimensionamento de lotes multiestágio, preservação da preparação, problema inteiro misto, metaheurística**.

Abstract

This paper considers the lot-sizing problem in a multilevel production system. The problem consists into determine a production plan which meets the demand of end products and its components in each period of a finite planning horizon, without backlogging, and minimizing the involved costs. In this paper, we consider the multilevel capacitated lot-sizing problem with setup carry-over, possibility of overtime utilization, positive setup costs and times and positive lead-time also. To solve this problem, we proposed a hybrid heuristic method based on the mathematical formulation of the problem (fix-and-optimize) with the bee algorithm metaheuristic. We made tests with some literature instances and the results are compared with another heuristic that approach the same problem. The results for the tested classes have proved promising.

KEYWORDS: **Multilevel lot-sizing, setup carry-over, mixed integer problem, metaheuristic**.

1. Introdução

O problema de dimensionamento de lotes consiste em determinar o tamanho dos lotes de cada produto que será produzido ao longo de um horizonte de planejamento com T períodos, com o intuito de suprir as demandas, sem atrasos. O objetivo do problema é minimizar os custos de preparações de máquinas, custos de estocagem e, em alguns casos, custos de produção e custos de horas extras.

Os problemas de dimensionamento de lotes podem ser divididos em monoestágio e multiestágio. O problema de dimensionamento de lotes monoestágio se caracteriza principalmente por ter produtos sendo produzidos em um mesmo meio de produção mas sem relação de produção entre eles, ou seja, a produção de um produto não depende da produção de outro produto. O problema de dimensionamento de lotes multiestágio se caracteriza principalmente por ter múltiplos produtos sendo produzidos em um mesmo meio de produção segundo uma estrutura de produto bem definida. Desta forma, alguns produtos têm sua produção dependendo da produção dos produtos sucessores.

O problema MLCLSP (problema de dimensionamento de lotes multiestágio com restrição de capacidades) se caracteriza como um problema onde os recursos utilizados na produção têm limites de capacidade. Além disso, outras características adicionais para o problema podem ser consideradas, tais como: tempos e custos de preparações positivos, possibilidade de preservar a preparação e um *lead-time*[1] positivo e fixo de um período, independente do produto. No caso da preservação ser considerada, somente uma preparação por recurso e período poderá ser preservada.

O MLCLSP foi inicialmente proposto em Billington et al. (1983). Neste artigo, os autores apresentam uma formulação matemática para o problema baseado na extensão da formulação do problema de dimensionamento de lotes monoestágio com restrição de capacidade. Deste então, o problema tem sido abordado na literatura, de inúmeras maneiras e com variadas características. Alguns trabalhos se destacam pela importância, como Maes et al. (1991), onde os autores provam que o problema de encontrar uma solução factível para o problema MLCLSP com tempo de preparação positivo é NP-Completo. Além disso, neste mesmo trabalho, os autores propõem algumas heurísticas baseadas em relaxação linear, o que até então não havia sido feito para este tipo de problema.

Outro trabalho importante na área é o de Stadtler (2003). Neste artigo, o autor apresenta uma reformulação do problema MLCLSP baseado em localização de facilidades. Esta reformulação é feita com o intuito de obter uma formulação matemática mais forte. Os autores utilizaram uma heurística de decomposição por janela de tempo para resolver o problema. As janelas de tempo consistem em conjuntos de 4 períodos, onde o último deles é resolvido de forma relaxada. A janela seguinte começa no terceiro período da anterior, ou seja, dois períodos são sobrepostos em janelas consecutivas e a decisão tomada na janela anterior é levada em consideração na janela seguinte. Neste trabalho possibilita-se a utilização de hora extra, porém a utilização desta resulta em um alto custo de produção.

Sahling et al. (2009) apresentam uma heurística do tipo *fix-and-optimize* para o problema de dimensionamento de lotes multiestágio com limitação da capacidade e preservação da preparação. Além disso, existe a possibilidade de utilização de hora extra e considera o *lead-time* positivo e igual a um para todos os componentes. A heurística se baseia no particionamento do conjunto de variáveis inteiras em subconjuntos disjuntos, com o intuito de reduzir o número de variáveis que serão otimizadas. Assim, a cada passo da heurística, uma partição fica "livre" para ser otimizada e as demais são fixadas. Neste artigo foram

[1] O *lead-time* é um tempo necessário para que um produto esteja pronto para o uso após o início de sua produção. Esse tempo é utilizado para evitar que um produto tenha sua produção iniciada antes que seus componentes tenham suas produções concluídas.

feitas quatro variações de decomposição para a escolha do conjunto de variáveis que são fixadas: decomposição orientada a produto, decomposição orientada a produto seguida de decomposição orientada a recurso, decomposição orientada a produto seguida de decomposição orientada a processo e decomposição orientada a produto, depois orientada a recurso e por último orientada a processo.

Em Helber e Sahling (2010) foi apresentada uma heurística do tipo *fix-and-optimize* para o problema MLCLSP, com possiblidade de utilização de hora extra, *lead-time* positivo, porém não existe a possibilidade de preservar a preparação. Neste trabalho, os autores utilizam instâncias de testes da literatura (geradas em Tempelmeier e Derstroff (1996) e Stadtler e Sürie (2000)) e comparam os resultados com as heurísticas propostas em Tempelmeier e Derstroff (1996) e Stadtler (2003).

Um método híbrido para o problema MLCLSP foi proposto por Almeder (2009). Neste trabalho, considera-se a possibilidade de utilização de hora extra, mas sem a possibilidade de preservação da preparação. O método híbrido proposto consiste em um procedimento misto de Colônia de Formigas (ACO) com ferramentas de solução de problemas LP/MIP, mais especificamente o pacote de otimização CPLEX. Neste caso, a ACO foi utilizada para determinar os períodos nos quais cada produto deveria ser produzido (as preparações). Baseado nestas decisões, as quantidades de cada produto a serem produzidas em cada período são determinadas pelo CPLEX. Os autores indicam que este método obteve bom desempenho para instâncias de tamanho pequeno e médio. Para instâncias de grande porte, encontrou soluções de alta qualidade, mas não melhores do que as obtidas pelas heurísticas especializadas para o problema (Stadtler (2003) e Pitakaso et al. (2006)).

Neste artigo apresentaremos um método híbrido para a solução do problema MLCLSP, proposto em Sahling et al. (2009), ou seja, com possibilidade de preservação da preparação, tempos e custos de preparações positivos, estoque inicial e *lead-time* positivo. O método híbrido se compõe da heurística *fix-and-optimize*, juntamente com algoritmo de abelhas que determina as variáveis de preparação e preservação que serão fixadas a cada iteração do método. No nosso método, utilizamos o algoritmo de abelhas para definir quais variáveis do conjunto de variáveis binárias devemos fixar e quais devemos otimizar. Desta forma, as partições são definidas como um conjunto de variáveis escolhidas segundo as formas de busca do algoritmo de abelhas.

O trabalho foi organizado da seguinte forma. Na Seção 2 são apresentadas a definição do problema e duas formulações matemáticas, a primeira é uma formulação agregada e a segunda modelagem consiste na reformulação da primeira por localização de facilidades. Na Seção 3 apresentam-se definições sobre a heurística de fixação *fix-and-optimize* e, a seguir, na Seção 4 descreve-se a metaheurística algoritmo de abelhas. Na Seção 5, o método híbrido proposto para a resolução do problema é apresentado. Os resultados computacionais são indicados na Seção 6, seguido das conclusões e perspectivas futuras na Seção 7.

2. Modelagem Matemática

Neste trabalho apresentaremos a formulação agregada proposta em Sahling et al. (2009) e a seguir o modelo reescrito segundo a reformulação por localização de facilidades proposta em Stadtler (2003).

O problema proposto em Sahling et al. (2009) e abordado neste trabalho tem as seguintes características:

- Demanda externa determinística;
- Estoque inicial;
- Recursos com limites de capacidade;
- Possibilidade de utilização de hora extra com um custo elevado;

- Custos e tempos de preparações positivos;
- Possibilidade de preservação das preparações entre períodos;
- *Lead-time* positivo e igual a um para todos os produtos.

Para a formulação agregada, considere o seguinte conjunto de índices, parâmetros e variáveis:

Conjuntos e índices	
N	Conjunto de produtos ($i \in \{1, \ldots, N\}$);
M	Conjunto de recursos ($m \in \{1, \ldots, M\}$);
T	Conjunto de períodos ($t \in \{1, \ldots, T\}$);
$S(i)$	Conjunto de sucessores de i;
$A(i)$	Conjunto de antecessores de i;
K_m	Conjunto de produtos produzindo com o recurso m.
Parâmetros	
S_i	Custo de preparação relacionada ao produto i;
h_i	Custo de estoque relacionado ao produto i;
oc_m	Custo de hora extra relacionada ao recurso m;
r_{ij}	Unidades do produto i necessários para produzir uma unidade do produto j;
d_{it}	Demanda externa do produto i no período t;
$\hat{I}_i$	Estoque inicial do produto i;
a_i	Tempo de produção de cada unidade do produto i;
ts_i	Tempo de preparação para a produção do produto i;
C_{mt}	Capacidade do recurso m no período t.
Variáveis	
O_{mt}	Quantidade utilizada de hora extra do recurso m no período t;
I_{it}	Estoque do produto i no final do período t;
I_{i0}	Estoque inicial do produto i;
	(estoque inicial $\hat{I}_i$ menos a demanda dependende do primeiro período)
Y_{it}	Variável de preparação/produção do produto i no período t;
	($Y_{it} = 1$, se $X_{it} > 0$
	$Y_{it} = 0$, caso contrário.)
W_{it}	Variável de preservação da preparação para produção de i no período t;
	($W_{it} = 1$, se houver preservação da preparação
	$W_{it} = 0$, caso contrário.)
V_{mt}	Variável auxiliar do recurso m no período t;
X_{it}	Quantidade produzida do produto i no período t;

Tabela 1: Notação utilizada na formulação matemática agregada.

Para apresentar a formulação matemática, considere também a demanda de escalão (D_{it}) que consiste na soma das demandas independentes (d_{it}) e dependentes ($\sum_{j \in S(i)} r_{ij} * D_{j,t+1}$) e dada por:

$$D_{it} = d_{i\pi} + \sum_{j \in S(i)} r_{ij} * D_{j,\pi+1}$$

Além disso, considere B_{it} que é a demanda de escalão acumulada do produto i dos período t até o último período do horizonte de planejamento (T). Desta forma a produção do produto i no período t pode ser limitada por este valor, pois a produção precisa ser no

máximo igual a demanda total do problema até o final do horizonte de planejamento. A demanda de escalão acumulada é dada por:

$$B_{it} = \max\{0, d_{it} + \sum_{j \in S(i)} r_{ij} * B_{j,t+1} - \max\{0, \hat{I}_i - \sum_{\pi=0}^{t-1}(d_{i\pi} + \sum_{j \in S(i)} r_{ij} * B_{j,\pi+1})\}\}$$

A formulação agregada proposta em Sahling et al. (2009) segue abaixo.

$$\text{minimize} \quad Z = \sum_{i=1}^{N}\sum_{t=1}^{T} h_i * I_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} s_i * (Y_{it} - W_{it}) + \sum_{m=1}^{M}\sum_{t=1}^{T} oc_m * O_{mt} \quad (1)$$

sujeito a:

$$I_{i,t-1} + X_{it} - \sum_{j \in S(i)} r_{ij} * X_{j,t+1} - I_{it} = d_{it} \qquad \forall i, t = 2, \ldots, T-1 \quad (2)$$

$$I_{i,T-1} + X_{iT} - I_{iT} = d_{iT} \qquad \forall i \quad (3)$$

$$\hat{I}_i - \sum_{j \in S(i)} r_{ij} * X_{j1} = I_{i0} \qquad \forall i \quad (4)$$

$$I_{i0} + X_{i1} - \sum_{j \in S(i)} r_{ij} * X_{j2} - I_{i1} = d_{i1} \qquad \forall i \quad (5)$$

$$\sum_{i \in K_m} [a_i * X_{it} + ts_i * (Y_{it} - W_{it})] \leq C_{mt} + O_{mt} \qquad \forall m, t \quad (6)$$

$$X_{it} \leq B_{it} * Y_{it} \qquad \forall i, t \quad (7)$$

$$\sum_{i \in K_m} W_{it} \leq 1 \qquad \forall m, t \quad (8)$$

$$W_{it} \leq Y_{i,t-1} \qquad \forall i, t = 2, \ldots, T \quad (9)$$

$$W_{it} \leq Y_{it} \qquad \forall i, t \quad (10)$$

$$W_{it} + W_{i,t+1} \leq 1 + V_{mt} \qquad \forall m, i \in K_m, t \quad (11)$$

$$(Y_{it} - W_{it}) + V_{mt} \leq 1 \qquad \forall m, i \in K_m, t \quad (12)$$

$$W_{i1} = 0 \qquad \forall i \quad (13)$$

$$W_{i,T+1} = 0 \qquad \forall i \quad (14)$$

$$X_{it} \geq 0 \qquad \forall i, t \quad (15)$$

$$I_{it} \geq 0 \qquad \forall i, t \quad (16)$$

$$I_{i0} \geq 0 \qquad \forall i \quad (17)$$

$$O_{mt}, V_{mt} \geq 0 \qquad \forall m, t \quad (18)$$

$$Y_{it}, W_{it} \in \{0, 1\} \qquad \forall i, t \quad (19)$$

A função objetivo (1) consiste da soma dos custos de estoque, de preparação e de hora extra. O custo de preparação é computado, para um determinado período, somente se houver preparação no período ($Y_{it} = 1$) e não houver preservação da preparação ($W_{it} = 0$). As restrições (2)-(5) são responsáveis pelo balanceamento do estoque, com um *lead-time*=1 somente para os produtos utilizados como componentes de sucessores e não para os que suprem demanda externa, ou seja, os produtos finais. As restrições de balanceamento de estoque foram separadas para um melhor entendimento. As restrições (3) tratam do balanceamento do último período do horizonte de planejamento, onde não existe demanda

dependente. As restrições (4) fazem o cálculo do estoque inicial resultante (estoque inicial menos os componentes utilizados na produção do primeiro período), estas são necessárias devido ao *lead-time* positivo. As restrições (5) e (2) são restrições de balanceamento de estoque clássicas e foram divididas apenas para deixar explícita a utilização do estoque inicial (I_{i0}) nas restrições (5).

As restrições (6) são responsáveis pela limitação da capacidade disponível além da hora extra necessária para factibilizar o plano de produção. As restrições (7) obrigam a necessidade de preparação caso haja produção no período, além de limitar a produção pela demanda de escalão acumulada do período atual até o horizonte de planejamento. Em (8) é garantido que haverá no máximo uma preservação da preparação por recurso em cada período do horizonte de planejamento. As restrições (9) e (10) garantem que para haver preservação da preparação da produção no período t deve existir preparação nos períodos $t-1$ e t.

As restrições (11) e (12) garantem a possibilidade de preservação da preservação por múltiplos períodos. Por exemplo, se preservarmos a preparação para um produto k do período $t-1$ para o período t, e se produzirmos somente este produto durante todo o período t, no recurso m, e preservarmos novamente a preparação para o período $t+1$, de acordo com (11) temos que $W_{kt} + W_{k,t+1} = 2$ e assim $2 \leq 1 + V_{mt}$. Como $V_{mt} \geq 0$, então $V_{mt} \geq 1$. Porém de acordo com (12) podemos verificar que $(Y_{kt} - W_{kt}) = 0$ devido a restrição (10), resultando em $V_{mt} \leq 1$. Como resultado temos que $V_{mt} = 1$. O valor de $V_{mt} = 1$ faz com que $Y_{it} - W_{it} = 0, \quad \forall i \in K_m$. Como somente uma preservação por período e recurso é permitida, todos $W_{it} = 0, \quad \forall i \in K_m, i \neq k$ e com isto $Y_{it} = 0, \quad \forall i \in K_m, i \neq k$. Desta forma garante-se que nenhum outro produto será produzido no período t utilizando o mesmo recurso do produto k.

As restrições (13) e (14) estabelecem as preservações das preparações do primeiro e do período seguinte ao final do horizonte de planejamento. As demais restrições são de domínio.

Com o objetivo de obter uma formulação que ofereça limitantes inferiores de melhor qualidade, o modelo (1)-(19) foi reformulado de acordo com a reformulação proposta em Stadtler (2003). Esta formulação será utilizada no algoritmo híbrido proposto.

A formulação utilizado em Sahling et al. (2009) foi o modelo (1) - (19) com reformulação por localização de facilidades apenas nas restrições (7). Desta forma, as restrições foram substituidas pelos conjuntos de restrições: $(X_{it} = \sum_{\pi=t}^{T} \delta_{it\pi} \quad \forall i,t)$ e $(\delta_{it\pi} \leq B_{i\pi} * Y_{it} \quad \forall i,t,\pi \geq t)$.

Para apresentar esta formulação, considere as seguintes variáveis auxiliares: $\delta_{it\pi}$ que indicam as quantidades produzidas de cada produto $i = 1,\ldots,N$ e em cada período $t = 1,\ldots,T$ para consumo nos períodos $\pi = t,\ldots,T$.

Estas variáveis substituem as variáveis de produção do modelo agregado da seguinte forma: $X_{it} = \sum_{\pi=t}^{T} \delta_{it\pi}$

O modelo reescrito segue abaixo.

$$\text{minimize} \qquad Z = \sum_{i=1}^{N}\sum_{t=1}^{T} hi * I_{it} + \sum_{i=1}^{N}\sum_{t=1}^{T} s_i * (Y_{it} - W_{it}) + \sum_{m=1}^{M}\sum_{t=1}^{T} oc_m * O_{mt} \qquad (20)$$

sujeito a: (21)

$(8) - (14)$

(16) – (19)

$$\sum_{\pi=1}^{t} \delta_{i\pi t} = D_{it} \qquad \forall i, t, D_{it} > 0 \tag{22}$$

$$I_{i,t-1} + \sum_{\substack{\pi=t \\ seD_{i\pi}>0}}^{T} \delta_{it\pi} - \sum_{j \in S(i)} \sum_{\substack{\pi=t+1 \\ seD_{i\pi}>0}}^{T} r_{ij} * \delta_{j,t+1,\pi} - I_{it} = d_{it} \qquad \forall i, t = 2, \ldots, T-1 \tag{23}$$

$$I_{i,T-1} + \delta_{\substack{iTT \\ seD_{iT}>0}} - I_{iT} = d_{iT} \qquad \forall i \tag{24}$$

$$\hat{I}_i - \sum_{j \in S(i)} \sum_{\substack{\pi=1 \\ seD_{i\pi}>0}}^{T} r_{ij} * \delta_{j1\pi} - I_{i0} = 0 \qquad \forall i \tag{25}$$

$$I_{i0} - \sum_{\substack{\pi=1 \\ seD_{i\pi}>0}}^{T} \delta_{i1\pi} - \sum_{j \in S(i)} \sum_{\substack{\pi=2 \\ seD_{i\pi}>0}}^{T} r_{ij} * \delta_{j2\pi} - I_{i1} = d_{i1} \qquad \forall i \tag{26}$$

$$\sum_{i \in K_m} \sum_{\substack{\pi=t \\ seD_{i\pi}>0}}^{T} a_i * \delta_{it\pi} + \sum_{i \in K_m} ts_i * (Y_{it} - W_{it}) \leq C_{mt} + O_{mt} \qquad \forall m, t \tag{27}$$

$$\delta_{it\pi} \leq B_{i\pi} * Y_{it} \qquad \forall i, t, \pi \geq t, D_{i\pi} > 0 \tag{28}$$

$$\delta_{it\pi} \geq 0 \qquad \forall i, t, \pi \geq t, D_{i\pi} > 0 \tag{29}$$

A função objetivo (20) consiste da soma dos custos de estoque, de preparação e de hora extra. As restrições (22) garantem o atendimento da demanda (demanda de escalão). As restrições (23)-(26) são responsáveis pelo balanceamento do estoque, com *lead-time*=1. As restrições (27) são responsáveis pela limitação da capacidade. As equações (28) fazem com que exista a preparação caso haja produção no período e limita esta produção pela demanda de escalão acumulada. As variáveis de produção (localização de facilidades) são reais e positivas (29).

3.Heurística de fixação (*fix-and-optimize*)

A heurística de fixação *fix-and-optimize* foi inicialmente proposta em Pochet e Wolsey (2006) com o nome de *exchange*. Está heurística foi proposta como uma melhoria da heurística *relax-and-fix* (Pochet e Wolsey; 2006). Posteriormente, em Sahling et al. (2009) e Helber e Sahling (2010) recebeu o nome de *fix-and-optimize*.

A heurística se baseia na formulação matemática do problema e na partição do conjunto de variáveis binárias do problema em subconjuntos disjuntos. Desta forma, o objetivo da heurística é reduzir a dificuldade de solução dos subproblemas reduzindo o número de variáveis binárias que são otimizadas. A heurística consiste em liberar uma partição para otimização e fixar as demais partições no valor da solução incumbente. A cada iteração da heurística pode-se melhorar o valor da solução incumbente. O processo será interrompido quando algum critério de parada for satisfeito. Desta forma os subproblemas resolvidos são também problemas inteiros mistos, mas com um número menor de variáveis inteiras e, portanto, "mais fáceis" de serem resolvidos.

Portanto o problema é resolvido iterativamente e a cada iteração:

- Uma partição é liberada para otimização;
- Todas as demais partições são fixadas no valor da solução incumbente;
- O problema resultante é resolvido até a otimalidade ou até que algum critério de parada pré-definido seja satisfeito;

- A solução é armazenada para a próxima iteração.

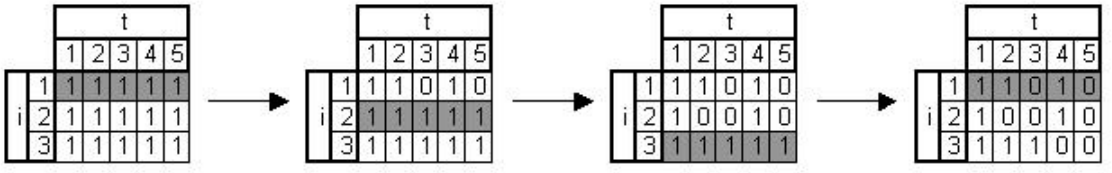

Figura 1: Exemplo de fixação das variáveis de preparação em um problema com 3 produtos e 5 períodos. (decomposição por produtos)

A Figura 1 mostra um exemplo bem simples do funcionamento da heurística *fix-and-optimize* onde i representa cada produto e t cada período do horizonte de planejamento. Cada tabela representa um passo da heurística e as setas indicam a ordem de precedência. Em cada iteração, os campos em cinza indicam as variáveis que estão livres para serem otimizados, enquanto as demais estão fixadas com os valores da solução da iteração anterior. Neste caso, a decomposição escolhida, foi por produtos, ou seja, a cada iteração um produto diferente é otimizado ao longo dos períodos.

Este tipo de abordagem, atrelado a forma como foi feita a escolha do conjunto de variáveis que são fixadas demonstrou resultados efetivos para os problemas considerados (Sahling et al. (2009) e Helber e Sahling (2010)).

4. Algoritmo de abelhas

A metaheurística algoritmo de abelhas foi apresentada em Pham et al. (2006). Esta metaheurística se baseia nos padrões de busca das abelhas por comida. O objetivo das abelhas, quando estão buscando comida é sempre maximizar a quantidade de comida adquirida dentro de um período, desta forma, são levados dois fatores em consideração na busca, distância da fonte de comida e facilidade de obter o alimento (quantidade de alimento).

Outro ponto importante na busca das abelhas por comida se encontra na forma como a busca é feita. No primeiro passo, as abelhas exploradoras são enviadas para encontrar os pontos de melhor coleta de alimentos, inicialmente de forma aleatória. Após o passo inicial, elas retornam, depositam o pólen recolhido e fazem uma dança conhecida por *waggle dance*. Esta dança é utilizada para passar as informações sobre o local onde cada abelha exploradora fez sua coleta de pólen. Com a dança, elas indicam a distância da fonte de comida, a direção na qual ela se encontra e a quantidade de comida existente. No segundo passo, são recrutadas as abelhas seguidoras, que buscarão a comida nos locais indicados pelas exploradoras. Os pontos mais promissores, ou seja, onde existe maior possibilidade de arrecadar comida, recebe um maior número de abelhas seguidoras. As informações sobre cada local de coleta dos alimentos é constantemente verificada, pois cada vez que as abelhas retornam à colméia, ocorre novamente a dança e outras abelhas seguidoras são enviadas para os locais promissores.

Dessa forma, o algoritmo é basicamente composto de duas formas de busca combinadas. As abelhas exploradoras fazem uma busca aleatória pelo espaço de solução do problema. Após isso, as abelhas seguidoras fazem uma busca local na vizinhança de cada um dos locais escolhidos para a verificação. Em seguida, são escolhidos os melhores indivíduos de cada vizinhança para formar a nova população de abelhas exploradoras, e o algoritmo se repete até que um critério de parada seja satisfeito.

5. Método híbrido proposto

Com o objetivo de resolver o problema descrito na Seção 2, desenvolveu-se um método híbrido utilizando algoritmo de abelhas e *fix-and-optimize*. No algoritmo *fix-and-optimize* de Sahling et al. (2009) as partições das variáveis binárias são obtidas utilizando-se decomposições por produto, por processo e por recurso.

No nosso algoritmo híbrido, utiliza-se o algoritmo de abelhas para determinar as partições do conjunto de variáveis binárias. No primeiro passo, as partições são geradas aleatoriamente (busca aleatória). No segundo passo, algumas variáveis são trocadas entre as partições, fazendo, desta forma, uma busca local na vizinhança e em seguida escolhendo o melhor particionamento para ser ponto de partida na próxima iteração.

Desta forma, cada abelha é definida pelas partições de fixação e matrizes para armazenar a solução do subproblema, substituindo as decomposições utilizadas em Sahling et al. (2009). A solução de cada abelha pode alterar o valor da solução incumbente, se for melhor.

Os subproblemas resultantes fixados do modelo (20)-(28) são resolvidos utilizando o programa ILOG CPLEX 12.1.

A estrutura básica da heurística proposta é apresentada no Algoritmo 1 e a notação utilizada na Tabela 2.

Conjuntos e índices	
$NT_{y,s}^{fix}$	Partição das variáveis de preparação fixadas no subproblema s;
$NT_{w,s}^{fix}$	Partição das variáveis de preservação da preparação fixadas no subproblema s;
$NT_{y,s}^{opt}$	Partição das variáveis de preparação livres no subproblema s;
$NT_{w,s}^{opt}$	Partição das variáveis de preservação da preparação livres no subproblema s;
Variáveis	
l	Contador de iterações;
Z, Z^{novo} e Z^{velho}	Valor da função objetivo (20) durante o processo iterativo;
$CapFeas$, $CapFeas^{novo}$	Indica se o subproblema e a solução incumbente utilizaram, ou não, hora extra;
$\overline{Y}_{it}$ e $\overline{W}_{it}$	Solução incumbente (preparação e preservação da preparação).

Tabela 2: Notação utilizada no Algoritmo 1.

No Algoritmo 1 as linhas de 1 a 3 definem a política de obtenção solução inicial, ou seja, uma solução de acordo com a política lote-por-lote. Nas linhas 5 e 6 armazena-se o restante da solução incumbente (preservação da preparação e valor da função objetivo). Nas linhas 7 e 8 verifica-se se houve a necessidade de uso de hora extra. Nas linhas 17 a 21 definimos as partições que serão fixadas e liberadas, resolvemos o subproblema e, em seguida, verificamos a utilização de hora extra. E nas linhas 22 a 26 armazenamos a solução encontrada no subproblema, caso essa seja melhor que a solução incumbente.

Ao final do algoritmo temos como resultado, a solução incumbente.

$\overline{Y}_{it} = 1 \qquad \forall (i,t) \in NT$;
$NT^{fix}_{y,s} \leftarrow NT$;
$NT^{fix}_{w,s} \leftarrow \emptyset$;
resolve o subproblema e determina o valor da função objetivo Z;
$Z^{novo} = Z$;
$\overline{W}_{it} = W_{it} \qquad \forall (i,t) \in NT$;
se *Uso de hora extra = 0* **então** CapFeas=verdadeiro
senão CapFeas=falso
gera a população inicial de abelhas exploradoras e resolve os subproblemas indicados por cada uma delas;
l=0;
repita
    l=l+1;
    $Z^{velho} = Z^{novo}$;
    para *cada abelha exploradora escolhida* **faça**
        recrute uma quantidade de abelhas seguidoras de acordo com a qualidade da solução do subproblema gerado pela abelha exploradora;
        para *cada abelha seguidora* **faça**
            determinar $NT^{opt}_{y,s}$ e $NT^{fix}_{y,s} = NT \setminus NT^{opt}_{y,s}$ para o subproblema;
            determinar $NT^{opt}_{w,s}$ e $NT^{fix}_{w,s} = NT \setminus NT^{opt}_{w,s}$ para o subproblema;
            resolve o subproblema e determina o valor da função objetivo Z;
            se *Uso de hora extra = 0* **então** $CapFeas^{novo} = verdadeiro$
            senão $CapFeas^{novo} = falso$
            se $Z < Z^{velho}$ *e* $\{CapFeas^{novo}$ *ou* $[not(CapFeas^{novo})$ *e* $not(CapFeas)]\}$ **então**
                $\overline{Y}_{it} = Y_{it} \qquad \forall (i,t) \in NT^{opt}_{y,s}$;
                $\overline{W}_{it} = W_{it} \qquad \forall (i,t) \in NT^{opt}_{w,s}$;
                $Z^{novo} = Z$;
                se $CapFeas^{novo}$ **então** $CapFeas = verdadeiro$
            fim
        fim
        escolhe a melhor abelha seguidora para substituir sua abelha exploradora;
    fim
até $l = l^{max}$ *ou* $Z^{novo} \geq Z^{velho}$;

Algoritmo 1: Pseudo-código da heurística híbrida proposta.

6. Testes e resultados computacionais

6.1. Instâncias de testes

As instâncias de testes utilizadas neste trabalho foram geradas em Tempelmeier e Buschkuhl (2009) e utilizadas por Sahling et al. (2009). Estas instâncias de testes foram divididas em 6 classes, de acordo com os números de produtos, períodos e recursos das instâncias, como pode ser visto na Tabela 3. Para a construção das instâncias de testes, os autores levaram em consideração seis propriedades importantes para este problema que são:

1. Tamanho do problema;
2. Estrutura de produto;
3. Perfil de demanda dos produtos finais;
4. Tempos de preparação;
5. A proporção entre custos de preparação e de estoque;
6. A capacidade utilizada.

6.2. Resultados computacionais

A heurística proposta foi implementada em C++ utilizando a biblioteca ILOG Concert e a ferramenta de resolução LP/MIP CPLEX para a solução dos subproblemas. Os testes

Classe	Nº Produtos	Nº Períodos	Nº Recursos	Qtde de Problemas
1	10	4	3	480
2	10	8	3	480
3	20	8	6	240
4	20	16	6	240
5	40	8	6	240
6	40	16	6	240

Tabela 3: Características das instâncias de testes

foram executados em um computador com processador Intel Core 2 Duo de 2.33 Ghz com 4MB de cache, e 3GB de memória RAM. O computador tem como sistema operacional o Debian GNU/Linux 4.0.

Como foi utilizada uma metaheurística probabilística para a definição das partições que serão fixadas e das partições livres, a heurística híbrida também se tornou probabilística, logo para verificarmos a qualidade da solução, cada instância de teste foi executada 10 vezes. Em seguida foi feita a média das soluções encontradas e a média dos tempos encontrados.

Na Tabela 4 são indicados a qualidade da solução e o tempo médio de execução das 6 classes de instâncias resolvidas pela heurística proposta, denotada pela sigla BFO. Além disso, a coluna FO indica a heurística de melhor qualidade dentre as heurística *fix-and-optimize* propostas em Sahling et al. (2009). Na melhor heurística, os autores utilizam decomposição por produto primeiro, seguida de decomposição por recurso e por último decomposição por processo. Como esta heurística não foi implementada e executada em uma máquina com as mesmas configurações da nossa, o tempo médio desta foi suprimido da Tabela 4, pois comparações entre os tempos não são apropriadas.

	Desvio (%)		Tempo (s)
Classe	FO	BFO	BFO
Classe 1	**0,12**	**0,12**	0,65
Classe 2	0,23	**0,22**	3,72
Classe 3	0,40	**0,23**	7,91
Classe 4	**0,14**	0,21	70,05
Classe 5	0,37	**0,32**	26,60
Classe 6	**0,19**	0,34	263,97

Tabela 4: Resultados computacionais.

7. Conclusões e perspectivas futuras

Como pode ser verificado, a heurística híbrida obteve valores, na média, bem semelhantes aos da heurística *fix-and-optimize* proposta por Sahling et al. (2009), demonstrando que o particionamento aleatório também gera bons resultados. Já o tempo computacional se mostrou bastante elevado, principalmente para as classes de instâncias maiores. Em Sahling et al. (2009) os tempos não aumentaram tanto, variando de 0,31s para a Classe 1 até 21,39s na Classe 6. O crescimento mais acentuado no tempo da heurística híbrida se deve ao aumento da dificuldade das instâncias e a sobrecarga (*overhead*) causada pela metaheurística.

Os resultados computacionais indicaram também dois outros fatores importantes. O primeiro foi o desvio padrão encontrado entre as 10 soluções de cada instância, que em alguns casos foi elevado. Isso demonstra instabilidade na heurística, o que indica a necessidade de

maiores ajustes para tornar o método mais robusto. O segundo fator foi o pequeno número de iterações da heurística na resolução de cada instância, que chegou a no máximo poucas dezenas. Desta forma, resolver instâncias de maior dificuldade também se faz necessário para que a sobrecarga causada pela metaheurística seja validada.

Além disso, a heurística proposta tem um conjunto de oito parâmetros que devem ser ajustados, desde o número de abelhas exploradoras até a quantidade de variáveis fixadas a cada subproblema. Desta forma novos estudos são necessários para ajustar estes parâmetros com as características do problema abordado e das especificidades das instâncias de testes.

Agradecimentos
Este trabalho teve o apoio do CNPq.

Referências bibliográficas

Almeder, C., (2010), A hybrid optimization approach for multi-level capacitated lot-sizing problems, European Journal of Operational Research, 200, 599-606.

Billington, P., McClain, J. and Thomas, L., (1983), Mathematical Programming Approaches to Capacity-Constrained MRP Systems: Review, Formulation and Problem Reduction, Management Science, 29, 1126-1141.

Helber, S. and Sahling, F., (2010), A fix-and-optimize approach for the multi-level capacitated lot sizing problem, International Journal of Production Economics, 123, 247 - 256.

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

Pham, D., Ghanbarzadeh, A. , Koç, E. , Otri, S. , Rahim, S. and Zaidi, M., (2006), The Bees Algorithm - A Novel Tool for Complex Optimisation Problems, Proceeding of IPROMS 2006 Conference, 2006, 454–461.

Pitakaso, R., Almeder, C., Doerner, K. and Hartl, R., (2006), Combining population-based and exact methods for multi-level capacitated lot-sizing problems, International Journal of Production Research, 44, 4755–4771.

Pochet Y. and Wolsey L., Production Planning by Mixed Integer Programming, Springer, Nova York, 2006.

Sahling, F., Buschkühl, L., Tempelmeier, H. and Helber, S., (2009), Solving a multi-level capacitated lot sizing problem with multi-period setup carry-over via a fix-and-optimize heuristic, Computers & Operations Research, 36, 2546–2553.

Stadtler, H., (2003), Multilevel Lot Sizing with Setup Times and Multiple Constrained Resources: Internally Rolling Schedules with Lot-Sizing Windows, Operations Research, 51, 487–502.

Stadtler,H. and Sürie, C. , (2000), Description of MLCLSP test instances, Technicalreport, Technische Universität Darmstadt.

Tempelmeier H. and Derstroff M., (1996), A Lagrangean-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times, Management Science, 42, 738–757.

Tempelmeier H. and Buschkuhl L. (2009), A heuristic for the dynamic multi-level capacitated lotsizing problem with linked lotsizes for general product structures, OR Spectrum, 31, 385–404.