PROGRAMAÇÃO MATEMÁTICA E BUSCAS LOCAIS APLICADAS AO PROBLEMA CAPACITADO DE DIMENSIONAMENTO DE LOTES MULTI-NÍVEL

Claudio Fabiano Motta Toledo
Departamento de Ciência da Computação - Universidade Federal de Lavras
Capus Universitário, C.P. 3037, CEP 37200-000, Lavras, MG
claudio@dcc.ufla.br

Renato Resende Ribeiro de Oliveira
Departamento de Ciência da Computação - Universidade Federal de Lavras
Capus Universitário, C.P. 3037, CEP 37200-000, Lavras, MG
renatorro@comp.ufla.br

Lucas de Oliveira
Departamento de Ciência da Computação - Universidade Federal de Lavras
Capus Universitário, C.P. 3037, CEP 37200-000, Lavras, MG
oliveiralukas@yahoo.com.br

Paulo Morelato França
Departamento de Matemática, Estatística e Computação - Universidade Estadual Paulista
R. Roberto Simonsen, 305, CEP 19060-900, P. Prudente, SP
paulo.morelato@fct.unesp.br

RESUMO

O presente artigo propõe buscas locais associadas à programação matemática na resolução do Problema Capacitado de Dimensionamento de Lotes Multi-Nível (PCDLMN). Um modelo matemático inteiro-misto é descrito para o problema bem como uma reformulação é apresentada. As buscas locais utilizadas são Busca Tabu e *Simulated Annealing*. Esses métodos fixam as variáveis binárias do modelo e um método exato é utilizado, em seguida, para determinar as variáveis contínuas no modelo reformulado. Os métodos propostos são avaliados em dois conjuntos de instâncias encontradas na literatura. Essas instâncias são solucionadas inicialmente utilizando o *solver* CPLEX. Os resultados das buscas locais são comparados àqueles retornados pelo *solver*. As abordagens apresentadas conseguem obter melhores resultados na maior parte das instâncias avaliadas.
PALAVRAS CHAVE. Programação da produção. Dimensionamento de lotes. Busca local. Area principal: Metaheurística.

ABSTRACT

The present paper proposes local search associated with mathematical programming to solve the Multi-Level Capacitated Lot Sizing Problem. A mixed-integer mathematical model is described for the problem as well as a reformulation is presented. The local searches used are Tabu Search and Simulated Annealing. These methods fix binary variables of the model and an exact method is used next to determine continuous variables for reformulated model. The proposed methods are evaluated on two set of instances found in literature. These instances are solved initially using the solver CPLEX. The local search results are compared with those returned by the solver. The approaches presented are able to found better results in the most part of instances evaluated.
KEYWORDS. Scheduling. Lot sizing. Local search. Main area: Metaheuristic.

1. Introdução

O presente artigo propõe um método híbrido para solucionar o Problema Capacitado de Dimensionamento de Lotes Multi-Nível (PCDLMN). O método combina meta-heurísticas de busca local e um *LP/MIP solver* exato (ILOG CPLEX) visando estabelecer a programação e o dimensionamento dos lotes de produção para problemas *big-bucket*. O objetivo é obter esse planejamento minimizando os custos de estoque e setup, respeitando também a capacidade das máquinas. O planejamento da produção em problemas *big-bucket* consiste em definir as quantidades de matéria-prima, itens intermediários e produtos finais que devem ser produzidos em cada periodo para se atender a demanda externa. Ao contrário de problemas do tipo *small-bucket*, não há uma discretização dos períodos em micro-períodos e a sequência de produção dentro de um período não importa.

As decisões tomadas em um processo de planejamento da produção são essenciais para as indústrias nos dias atuais, já que minimizar custos pode ser um fator crucial na manutenção da competitividade de seus produtos no mercado. Planejar a produção contemplando multiplos níveis é um processo complexo. Há uma interdependência entre as quantidades produzidas para diversos itens que dificulta sobremaneira as decisões envolvidas no planejameno da produção. O PCDLMN foi provado ser NP-Díficil por Maes *et al* (1991). Stadtler (1996) desenvolveu modelos matemáticos inteiro-misto para o PCDLMN dinâmico. Mesmo com o crescimento do poder computacional e o avanço dos *solvers* exatos, a resolução de instâncias industriais na otimalidade continua como um problema inviável em muitos casos.

Por isso, métodos heurísticos costumam ser propostos para tratar esse problema. Jans e Degraeve (2007) apresentam uma revisão bibliográfica sobre a utilização de meta-heurísticas em problemas de dimensionamento de lotes e programação da produção. Neste estudo foi desenvolvido um método híbrido que utiliza Busca Tabu (BT) e *Simulated Annealing* (SA) na determinação das variáveis binárias do modelo matemático do PCDLMN. Em seguida, o pacote CPLEX é utilizado para determinar as variáveis contínuas do problema. Para isso, uma reformulação do modelo original é proposta. O método BT foi proposto por Glover e Laguna (1997) e utiliza uma memória de curto prazo para armazenar soluções selecionadas durante o processo de busca em vizinhança. O método SA simula o processo de arrefecimento dos metais, onde existe a probabilidade de se aceitar soluções piores que a atual com o objetivo de escapar de mínimos locais (Kirkpatrick *et al*, 1983).

Han *et al* (2009) propõem um algoritmo de otimização por enxame de particulas. Operadores matemáticos são redefinidos para resolver um problema de dimensionamento de lotes multi-nível não capacitado. Os resultados obtidos são comparados aqueles encontrados por um algoritmo genético. Özdamar e Barbarosoglu (2000) utilizam uma divisão do PCDLMN em subproblemas menores que são solucionados usando simulated annealing integrado a um método de relaxação lagrangeana. Stadtler (2003) propõe uma decomposição do PCDLMN em um número reduzido de períodos. Cada subproblema é resolvido avançando com uma janela de tempo que retira os primeiros períodos e adiciona os subsequentes. Akartunali e Miller (2009) utilizam uma decomposição similar combinada com uma heurística do tipo *relax-and-fix* aplicada a uma reformulação fortalecida. Sahling *et al* (2009) resolve uma variação do PCLDMN através de uma heurística *fix-and-optimize*. O método fixa grande parte das variáveis binária do modelo matemático e utiliza um *MIP solver* para otimizar o modelo inteiro-misto reduzido. A mesma ideia é aplicada por Helber e Sahling (2010) para o problema original do PCDLMN. Uma abordagem híbrida proposta foi utilizada por Toledo *et al* (2009) em um algoritmo genético multi-populacional. O algoritmo genético define as variáveis binárias para um modelo integrado de dimensionamento de lotes e programação da produção aplicado à produção de bebidas. Almeder (2010) também utiliza esta idéia em um PCDLMN, mas definindo as variáveis binárias do modelo matemático utilizando um algoritmo do tipo colônia de formigas.

O artigo está dividido em 5 seções. O modelo matemático original e a reformulação proposta são apresentados na seção 2. A seção 3 contém a descrição dos métodos e seus respectivos pseu-

docódigos. A seção 4 apresenta os resultados computacionais. A conclusões obtidas neste trabalho estão na seção 5.

2. Modelo Matemático do PCDLMN

O PCDLMN consiste em minizar os custos totais que incluem os custos de estoque, custos de ajuste e penalidades por estouro da capacidade disponível. Essa minimização deve ser feita respeitando os limites de capacidade das máquinas, restrições de balanço de estoque e atribuição de produtos em cada período. Um modelo matemático será apresentado para descrever os objetivos e restrições do problema. Também é proposta uma reformulação desse modelo matemático, visando remover variáveis binárias já que elas serão determinadas pelas heurísticas.

2.1. Modelo Original

O modelo apresentado é o mesmo descrito em Stadtler (2003). As notações que serão utilizadas na definição do modelo são as seguintes:

Parâmetros:

J	Quantidade total de produtos.
T	Quantidade total de períodos.
M	Quantidade total de máquinas.
a_{mj}	Capacidade gasta para produzir uma unidade do produto j na máquina m.
B_{jt}	Limitante superior para tamanho do lote do produto j no período t.
C_{mt}	Capacidade total da máquina m no período t.
h_j	Custo de estoque de uma unidade do produto j em um período.
oc_{mt}	Custo por unidade de capacidade ultrapassada(*overtime*) na máquina m no período t.
P_{jt}	Demanda primária (externa) do produto j no período t.
r_{jk}	Quantidade do produto j necessária para produzir uma unidade de seu sucessor imediato k.
sc_j	Custo de ajuste do produto j.
st_{mj}	Gasto de capacidade para ajuste do produto j na máquina m (*setup time*).

Conjuntos:

S_j	Conjunto dos sucessores imediatos do produto j.

Variáveis:

x_{jt}	Quantidade produzida do produto j no período t.
y_{jt}	Variável de ajuste do produto j no período t.
i_{jt}	Quantidade estocada do produto j no período t.
z_{mt}	Quantidade de capacidade ultrapassada (*overtime*) na máquina m no período t.

A formulação matemática do PCDLMN cosiderando violação da capacidade (*overtime*) é descrita abaixo:

$$minimizar \sum_{j=1}^{J}\sum_{t=1}^{T}(h_{jt}\cdot i_{jt}) + \sum_{j=1}^{J}\sum_{t=1}^{T}(sc_{jt}\cdot y_{jt}) + \sum_{m=1}^{M}\sum_{t=1}^{T}(oc_{mt}\cdot z_{mt}) \tag{1}$$

sujeito à:

$$i_{jt-1} + x_{jt} = P_{jt} + i_{jt} + \sum_{k\in S_j}(r_{jk}\cdot x_{kt}) \qquad \forall j\in[1,J], t\in[1,T] \tag{2}$$

$$\sum_{j=1}^{J}(a_{mj}\cdot x_{jt}) + \sum_{j=1}^{J}(st_{mj}\cdot y_{jt}) \leq C_{mt} + z_{mt} \qquad \forall m\in[1,M], t\in[1,T] \tag{3}$$

$$x_{jt} \leq B_{jt} \cdot y_{jt} \qquad \forall j \in [1,J], t \in [1,T] \tag{4}$$

$$y \in \{0,1\}^{JxT} \tag{5}$$

$$i \geq 0^{JxT} \tag{6}$$

$$x \geq 0^{JxT} \tag{7}$$

$$z \geq 0^{MxT} \tag{8}$$

A função objetivo (1) do PCDLMN minimiza a soma dos custos de estoque, custos de ajuste (*setup*) e custos de violação da capacidade para todos os produtos, máquinas e períodos. A restrição de balanço de estoque (2) do problema considera a demanda interna e externa do produto. A quantidade produzida no período atual mais a quantidade estocada no período anterior devem atender as demandas primária e interna mais o estoque do produto no período atual. O estoque no período zero é o estoque inicial do produto e o estoque no período T é o estoque final. A restrição de capacidade da máquinas (3) determina o valor da capacidade violada em cada período. Essa restrição garante que o tempo de produção de todos os produtos mais o tempo de ajuste desses produtos, considerando mesma máquina e período, devem ser menor ou igual a capacidade disponível mais a capacidade violada. A restrição de ajuste 4 estabelece que os produtos devem estar ajustados, $y_{jt} = 1$, apenas quando houver produção, $x_{jt} > 0$. Os domínios das variáveis estão apresentados nas demais restrições.

2.2. Reformulação Proposta

O método criado consiste em fixar os valores da variável binária y_{jt} para que o modelo resultante se torne contínuo. Após fixar os valores binários, o *solver* CPLEX é executado para definir os lotes de produção e as quantidades estocadas. Uma reformulação foi proposta visando remover a variável y_{jt}. Inicialmente, alteramos a função objetivo (equação 1) para:

$$minimizar \sum_{j=1}^{J}\sum_{t=1}^{T}(h_{jt} \cdot i_{jt}) + \sum_{m=1}^{M}\sum_{t=1}^{T}(oc_{mt} \cdot z_{mt}) \tag{9}$$

O valor obtido irá retornar a soma dos custos de estoque e violação da capacidade. A meta-heurística fica encarregada de calcular os custo de ajuste que é adicionado ao valor retornado pelo *solver*. A restrição de balanço de estoque (2) fica inalterada, mas a restrição de capacidade (3) é substituída por:

$$\sum_{j=1}^{J}(a_{mj} \cdot x_{jt}) - z_{mt} \leq C_{mt} \qquad \forall m \in [1,M], t \in [1,T] \tag{10}$$

Na nova restrição de capacidade, o valor de C_{mt} é modificado constantemente pela meta-heurística que subtrai os tempos de ajustes da capacidade total da máquina m no período t. A equação (4) é eliminada do modelo. Para que a variável x_{jt} respeite os ajustes definidos pela meta-heurística, os limitantes superiores e inferiores da variável x_{jt} são ajustados como zero quando não há ajuste do produto j no período t ($y_{jt} = 0$). Quando há tal ajuste, o limitante superior é definido como infinito. Dessa forma, o modelo resultante é composto pelas equações (2), (6), (7), (8), (9) e (10).

3. Métodos de Resolução

Duas meta-heurísticas são propostas: Busca Tabu (BT) e *Simulated Annealing* (SA). A mesma codificação da solução é utilizada nas duas meta-heurísticas. Trata-se de uma matriz do tipo produtos por períodos, JxT, com entradas binárias. As entradas com valor 1 indicam ajuste do produto

em determinado período. A Figura 1 apresenta uma possível codificação da solução considerando 4 produtos e 6 períodos.

	T_1	T_2	T_3	T_4	T_5	T_6
P_1	1	1	1	0	0	1
P_2	1	0	0	0	1	1
P_3	0	1	0	1	0	1
P_4	1	0	1	0	1	0

Figura 1. Exemplo de codificação de solução com 4 produtos e 6 períodos.

Na Figura 1 todos os produtos, exceto P_3, serão produzidos no período T_1. A partir da codificação apresentada, o cálculo da função de *fitness* é realizado seguindo os passos descritos na Figura 2. Os custos de ajuste e os tempos de ajuste são calculados a partir da codificação da solução (Passo 1). As capacidades das máquinas (Passo 2) e os limitantes das variáveis x_{jt} (Passo 3) são redefinidos a partir da codificação e serão utilizados pelo modelo reformulado. O CPLEX é então executado para o modelo reformulado (Passo 4) e o valor final da solução é calculado (Passo 5). Esse valor é composto por aquele obtido pelo solver mais os custos de ajuste anteriormente calculados.

Passo 1: Calcular custos totais de ajuste de acordo com os valores contidos na codificação.
Passo 2: Atualizar variável C_{mt} do modelo, subtraindo da capacidade total da máquina m no período t os tempos de ajuste de acordo com os valores contidos na codificação.
Passo 3: Atualizar os limitantes superiores da variável x_{jt} do modelo de acordo com os valores contidos na codificação.
Passo 4: Executar ILOG CPLEX e receber o valor da função objetivo (equação 9).
Passo 5: Somar o valor da função objetivo retornada pelo *solver* aos custos de ajuste. Retornar resultado como valor de *fitness*.

Figura 2. Cálculo de fitness da solução.

As soluções iniciais para os métodos são criadas de forma trivial para que, em seguida, sejam eliminados os ajustes inúteis. Isso é feito ajustando toda a matriz de codificação com o valor 1 seguido pelo cálculo do valor dessa solução trivial. A solução obtida pelo CPLEX irá fornecer valores para x_{jt}. Em seguida, o procedimento de inicialização ajusta as entradas $S[j,t] = 0$ na matriz de codificação S quando $x_{jt} = 0$. A Figura 3 apresenta o pseudocódigo do procedimento de inicialização.

```
Procedimento Inicializa(S)
    Ajusta toda codificação de S para 1;
    CalculaFitness(S);
    x ← Matriz com valor dos lotes gerados pelo CPLEX;
    Para j ← 1 até J faça
        Para t ← 1 até T faça
            Se (x[j,t] = 0) então
                S[j,t] ← 0;
        Fim para;
    Fim para;
    Recalcula custos de ajuste de S;
Fim Inicializa;
```

Figura 3. Pseudocódigo do procedimento Inicializa()

Quatro tipos de movimentos de busca em vizinhança foram definidos para as meta-heurísticas propostas. Eles são detalhados na próxima subseção.

3.1. Movimentos Locais

O primeiro movimento local consiste em sortear uma posição da matriz aleatoriamente e inverter o valor binário encontrado. A Figura 4 ilustra o processo.

$$\begin{array}{|cccccc|}\hline 1&1&1&0&0&1\\ 1&0&0&0&1&1\\ 0&1&\underbrace{0}&1&0&1\\ 1&0&1&0&1&0\\ \hline\end{array}\begin{array}{|cccccc|}\hline 1&1&1&0&0&1\\ 1&0&0&0&1&1\\ 0&1&\underbrace{1}&1&0&1\\ 1&0&1&0&1&0\\ \hline\end{array}$$

Figura 4. Movimento Local 1

$$\begin{array}{|cccccc|}\hline 1&1&\underbrace{1}&0&0&1\\ 1&0&0&0&1&1\\ 0&1&\underbrace{0}&1&0&1\\ 1&0&1&0&1&0\\ \hline\end{array}\begin{array}{|cccccc|}\hline 1&1&\underbrace{0}&0&0&1\\ 1&0&0&0&1&1\\ 0&1&\underbrace{1}&1&0&1\\ 1&0&1&0&1&0\\ \hline\end{array}$$

Figura 5. Movimento Local 2

$$\begin{array}{|cccccc|}\hline 1&1&1&0&0&1\\ 1&0&0&0&1&1\\ 0&1&\underbrace{0}&1&0&\underbrace{1}\\ 1&0&1&0&1&0\\ \hline\end{array}\begin{array}{|cccccc|}\hline 1&1&1&0&0&1\\ 1&0&0&0&1&1\\ 0&1&\underbrace{1}&1&0&\underbrace{0}\\ 1&0&1&0&1&0\\ \hline\end{array}$$

Figura 6. Movimento Local 3

O segundo e terceiro movimento são bastante semelhantes. O segundo sorteia um período e, dentro deste período, sorteia dois produtos distintos. Os produtos têm seus valores trocados (Figura 5). O terceiro funciona da mesma maneira, só que sorteia um produto e, dentro da programação deste produto, sorteia dois períodos distintos trocando seus valores (Figura 6).

O quarto movimento utiliza o primeiro movimento. Ele sorteia um número entre 1 e o número total de produtos J. Em seguida, o primeiro movimento é executado tantas vezes quanto o número sorteado. O pseudocódigo deste movimento é descrito na Figura 7.

3.2. Busca Tabu

O pseudocódigo da Busca Tabu (BT) é descrito na Figura 8. O primeiro parâmetro do método é o número de iterações para executar a busca em vizinhança da solução atual $itrMax$. Esse valor é ajustado em 40 nos testes realizados. O número de movimentos locais que serão utilizados é definido por $nMovs$ que vale 3 nos testes realizados. Portanto, a BT só utiliza os três primeiros movimentos definidos na seção anterior. A variável $tamanhoListaTabu$ define o número máximo de movimentos na lista tabu. A lista tabu é dinâmica e ajustada com valor inicial 1 de $tamanhoListaTabu$. A solução S_{global} guarda a melhor solução encontrada durante toda execução.

O procedimento Movimenta() executa um dos movimentos na solução. O movimento m é sorteado dentre os $nMovs$ disponíveis que não estão na lista tabu. O procedimento Insere() coloca o movimento, passado por parâmetro, no final da lista tabu. O procedimento Atualiza() tem 50% de chance de modificar o valor de $tamanhoListaTabu$. Se isso ocorre, é sorteado um número entre 1 e $nMovs - 1$ como novo tamanho da lista tabu. Se o tamanho atual for menor que o anterior, os movimentos mais antigos na lista são liberados. Na prática, a lista tabu oscila entre manter um ou dois movimentos tabu.

```
Procedimento MovimentoLocal4(S)
    nMovimentos ← valor randômico de 1 a J;
    Para i ← 1 até nMovimentos faça
        MovimentoLocal1(S);
    Fim para;
Fim MovimentoLocal4;
```

Figura 7. Pseudocódigo do Movimento Local 4

```
BuscaTabu()
    nMovs ← 3;
    itrMax ← 40;
    tamanhoListaTabu ← 1;
    Inicializa(S_atual);
    f(S_global) ← ∞;
    listaTabu ← ∅;
    Enquanto (Tempo total não atingido) faça
        f(S_local) ← ∞;
        Para iteracoes ← 0 até itrMax faça
            S_aux ← S_atual;
            Movimenta(S_aux,m); // Onde m é um movimento aleatório onde m ∉ listaTabu.
            CalculaFitness(S_aux);
            Se (f(S_aux) < f(S_local)) então
                S_local ← S_aux;
                mBest ← m;
        Fim para;
        S_atual ← S_local;
        Se (f(S_atual) < f(S_global)) então
            S_global ← S_atual;
        Insere(listaTabu,mBest);
        Atualiza(listaTabu);
    Fim enquanto;
Fim BuscaTabu;
```

Figura 8. Pseudocódigo da Busca Tabu

3.3. *Simulated Annealing*

O pseudocódigo do *Simulated Annealing* é descrito na Figura 9. O parâmetro $itrMax$ define o número de movimentos realizados antes que a temperatura diminua. O valor utilizado para este parâmetro nos testes computacionais é 40 e a temperatura inicial é ajustada para 10000. O fator de arrefecimento da temperatura α foi fixado em 0,9, ou seja, a nova temperatura será 90% da temperatura anterior. O limite da temperatura foi fixado em 0,001, portanto toda vez que a temperatura chega abaixo desse valor ela é reaquecida e volta ao valor inicial.

```
SimulatedAnnealing()
    itrMax ← 40;
    α ← 0.9;
    Temp ← 10000;
    Inicializa(S);
    f(S_global) ← ∞;
    Enquanto (Tempo total não atingido) faça
        Para iteracoes ← 1 até itrMax faça
            S_aux ← S;
            Movimenta(S_aux);
            CalculaFitness(S_aux);
            δ ← (f(S_aux) − f(S));
            Se (δ < 0) então
                S ← S_aux;
                Se (f(S) < f(S_global)) então
                    S_global ← S;
            Senão
                Sorteia x ∈ [0, 1];
                Se (x < e^(−δ/Temp)) então
                    S ← S_aux;
        Fim para;
        Temp ← Temp ∗ α;
        Se (Temp < 0.001) então
            Temp ← 10000;
    Fim enquanto;
Fim SimulatedAnnealing;
```

Figura 9. Pseudocódigo do *Simulated Annealing*

A variável $Temp$ guarda o valor atual da temperatura, S guarda a solução atual e S_{global} guarda a melhor solução encontrada ao longo de toda a execução. O movimento aplicado em Movimenta() é sorteado dentre os quatro movimentos definidos anteriormente. O valor de δ é negativo, se houve melhora da solução, e positivo, se houve piora. Caso seja negativo, a nova solução substitui a solução corrente em S. Caso contrário, existe uma probabilidade de aceitação da nova solução, em função da temperatura e do valor de δ, mesmo sendo pior que a atual. Isso ajuda o método a escapar de mínimos locais.

Na seção seguinte serão descritos os testes realizados e os resultados obtidos tanto pela BT quanto pelo SA.

4. Resultados Computacionais

Os métodos desenvolvidos neste artigo foram testados em um total de 40 instâncias, sendo 20 delas instâncias que utilizam estrutura de produtos *general* e outras 20 que utilizam estrutura *assembly*. A estrutura *general* permite vários predecessores e vários sucessores para cada produto. A estrutura *assembly* permite que um produto tenha vários predecessores, mas apenas um sucessor. Essas instâncias foram propostas por Tempelmeier e Derstroff (1996) e Stadtler (2003), onde são

definidas como conjuntos A+ e C, respectivamente. O conjunto A+ agrupa instâncias com 10 produtos, 24 períodos e 3 máquinas. O conjunto C agrupa instâncias com 40 produtos, 16 períodos e 6 máquinas. Foram gerados 20 exemplares em cada conjunto, sendo 10 com estrutura *general* e outros 10 com estrutura *assembly*. Nenhum dos dois conjuntos considera tempos de ajuste (*setup times*).

Para avaliar o desempenho dos métodos, uma comparação é feita com os resultados obtidos pelo *MIP solver* CPLEX 11.1 quando soluciona os conjuntos A+ e C utilizando o modelo original do PCDLMN (seção 2.1). Os métodos foram executados 10 vezes em cada instância com tempo de execução de 180 segundos no grupo A+ e 500 segundos no grupo C. O CPLEX foi executado uma vez para cada instância utilizando o mesmo tempo de execução das metaheurísticas. Todos os testes foram realizados em um processador Intel Core 2 Duo 2,66 GHz e 2 GB de Memória RAM. A Tabela 1 apresenta os resultados obtidos para o conjunto A+. A letra G no nome das instâncias indica que ela utiliza estrutura de produtos *general*, já a letra K indica utilização de estrutura *assembly*. Os melhores resultados para cada instância estão destacados.

Instância	CPLEX	Média BT	Desvio(%) BT	Média SA	Desvio(%) SA
AG501130	155.310	**153.682**	-1,05	154.492	-0,53
AG501131	147.580	**146.387**	-0,81	149.688	1,43
AG501132	155.669	156.327	0,42	**155.354**	-0,20
AG501141	**171.122**	178.464	4,29	179.302	4,78
AG501142	**193.427**	197.225	1,96	196.355	1,51
AG502130	167.378	**166.492**	-0,53	167.184	-0,12
AG502131	**146.393**	147.171	0,53	149.636	2,22
AG502141	**178.299**	178.689	0,22	182.246	2,21
AG502232	**119.064**	120.995	1,62	122.309	2,73
AG502531	**130.481**	130.917	0,33	133.027	1,95
AK501131	130.383	126.555	-2,94	**126.338**	-3,10
AK501132	123.945	**122.940**	-0,81	123.318	-0,51
AK501141	177.300	176.324	-0,55	**175.061**	-1,26
AK501142	163.695	**163.561**	-0,08	164.503	0,49
AK501432	112.543	**109.001**	-3,15	110.397	-1,90
AK502130	132.413	**129.891**	-1,90	130.744	-1,26
AK502131	119.606	**115.028**	-3,83	115.801	-3,18
AK502132	117.482	**117.084**	-0,34	118.439	0,81
AK502142	153.105	151.716	-0,91	**150.936**	-1,42
AK502432	108.910	**105.639**	-3,00	107.941	-0,89

Tabela 1. Resultados obtidos para o conjunto A+.

A Tabela 1 lista os valores obtidos pelo *solver* CPLEX e o valor médio retornado pelos métodos propostos. Esse valor médio é calculado considerando a solução final obtida em cada uma das 10 execuções. Também são apresentados os devios dos métodos em relação às soluções do *solver*. Esse desvio é calculado como $Desvio(\%) = 100 \cdot \frac{S-S^*}{S^*}$, onde S^* é a solução retornada pelo CPLEX e S é o valor médio encontrado pelos métodos BT ou SA.

A BT conseguiu 14 resultados que na média foram melhores que a solução encontrada pelo CPLEX. O SA obteve 11 resultados que na média foram melhores que os obtidos pelo CPLEX. Nas demais instâncias, BT e SA apresentaram desvios máximos de 4,29% e 4,78%, respectivamente. Esses desvios foram alcançados ambos na instância AG501141. Todavia, o desvio médio obtido no conjunto A+ pela BT ficou em torno de -0,53% e pelo SA em torno de 0,19%. Isso mostra que na média a BT conseguiu uma melhoria acima de 0,5% em relação ao CPLEX. Levando em conta

os melhores resultados obtidos pelos três métodos, a BT conseguiu o melhor resultado médio em 10 das 20 instâncias e o SA conseguiu o melhor resultado em 4 das 20 instâncias. Nas outras 6 instâncias o CPLEX obteve a melhor solução. A Tabela 2 mostra os resultados obtidos no conjunto de instâncias C.

Instância	CPLEX	Média BT	Desvio(%) BT	Média SA	Desvio(%) SA
CG501120	1.334.857	**1.293.870**	-3,07	1.307.280	-2,07
CG501121	1.356.552	**1.299.470**	-4,21	1.322.260	-2,53
CG501122	**1.803.171**	1.851.740	2,69	1.826.820	1,31
CG501131	632.999	**619.172**	-2,18	626.545	-1,02
CG501132	**841.998**	866.637	2,93	859.786	2,11
CG501141	821.679	**818.485**	-0,39	825.400	0,45
CG501142	**1.160.142**	1.180.380	1,74	1.175.410	1,32
CG501222	956.113	863.425	-9,69	**857.563**	-10,30
CG502221	996.591	905.209	-9,17	**900.021**	-9,69
CG502222	989.985	**883.303**	-10,78	888.936	-10,21
CK501120	184.098	179.849	-2,31	**179.775**	-2,35
CK501121	177.044	**172.741**	-2,43	173.004	-2,28
CK501122	**206.112**	211.830	2,77	216.973	5,27
CK501132	**97.982**	100.257	2,32	101.307	3,39
CK501142	121.319	**119.145**	-1,79	119.922	-1,15
CK501221	131.441	**118.877**	-9,56	121.434	-7,61
CK501222	125.535	**117.372**	-6,50	119.309	-4,96
CK501422	132.469	**121.563**	-8,23	124.218	-6,23
CK502221	131.595	**121.583**	-7,61	123.451	-6,19
CK502222	129.834	**117.946**	-9,16	120.228	-7,40

Tabela 2. Resultados obtidos para o conjunto C.

A BT obteve 15 resultados que na média superam aqueles obtidos pelo CPLEX. O SA obteve 14 resultados que superam o CPLEX na média das execuções. Nos demais resultados BT e SA apresentaram desvios máximos de 2,93% e 5,27%, respectivamente. Esses desvios foram alcançados nas instâncias CG501132 (BT) e CK501122 (SA). Entretanto os desvios médios obtidos por BT e SA ficaram em torno de -3,73% (BT) e -3,01% (SA). Isso mostra que na média os dois métodos conseguiram melhoria acima de 3% sobre as soluções do CPLEX. Considerando novamente apenas os melhores resultados dos três métodos, a BT conseguiu o melhor resultado em 12 das 20 instâncias e o SA conseguiu o melhor resultado em 3 das 20 instâncias. Nas outas 5 instâncias o CPLEX obteve o melhor resultado.

Deve-se observar que os resultados médios obtidos pelos métodos propostos, considerando 10 execuções, estão sendo comparados ao *solver* CPLEX. Porém, uma outra avaliação possível consiste em considerar o número de vitórias, ou seja, o total de execuções em que BT e SA superaram a solução obtida pelo CPLEX ao final de cada execução. Um total de 200 execuções foram realizadas por cada método em cada conjunto de instância. A Figura 10 apresenta o percentual de vitórias obtidas por cada método em cada conjunto dentro dessas 200 execuções. BT conseguiu o melhor desempenho no conjunto A+. No conjunto C, os dois métodos conseguiram exatamente o mesmo numero de vitórias em relação ao CPLEX nas 200 execuções. Podemos observar que o percentual de vitórias das metaheurísticas aumenta consideravelmente no conjunto C. Isso permite concluir que o desempenho das metaheurísticas é melhor no conjunto com instâncias mais complexas.

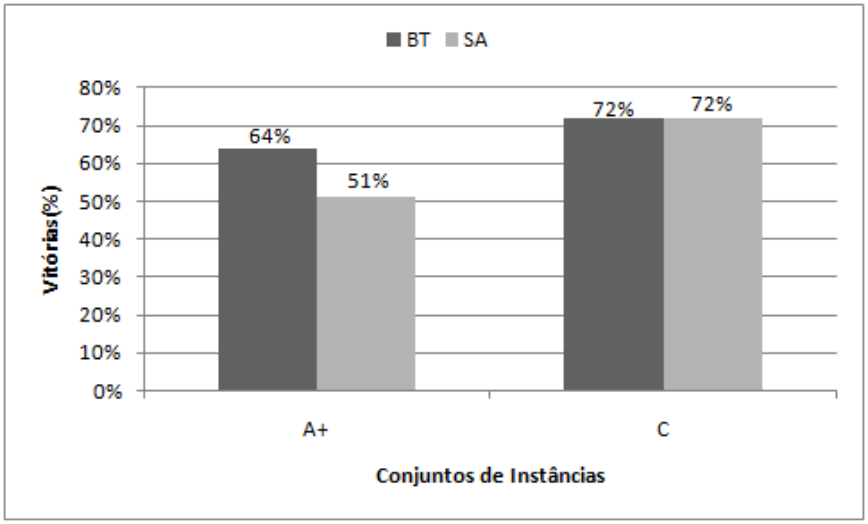

Figura 10. Vitórias sobre o CPLEX.

5. Conclusão

O presente estudo propõe duas metaheurísticas de busca local para solucionar o Problema Capacitado de Dimensionamento de Lotes Multi-Nível (PCDLMN). Os métodos associam as metaheurística Busca Tabu (BT) e *Simulated Annealing* (SA) a um *MIP/LP solver*, onde BT e SA determinam as variáveis binárias (atribuição de produtos a períodos) e o *solver* soluciona de forma exata um modelo contínuo (determinação dos lotes). Esse modelo é uma reformulação do modelo inteiro-misto apresentado para o PCDLMN. Os métodos propostos diferem pela forma como realizam a busca local sobre uma solução corrente. Dois conjuntos de instâncias testes obtidos da literatura são utilizados nos testes computacionais. Os resultados da metaheurísticas são comparados às soluções encontradas usando CPLEX. Ambos os métodos conseguem desvios percentuais baixos em relação às soluções obtidas pelo CPLEX, superando a solução final do método exato diversas vezes. O método BT com a configuração de parâmetros usada demonstrou ser mais adequado para solucionar instâncias simples, superando na média os resultados obtidos pelo CPLEX na maioria das instâncias. A BT obteve uma melhoria média acima de 0,5% em relação ao CPLEX. Nas instâncias de maior porte, BT também mostrou ter um desempenho superior. Entretanto, SA obtém um desempenho muito próximo ao da BT. Os dois métodos conseguem na média superar os resultados obtidos pelo CPLEX, obtendo ambos uma melhoria acima de 3%. Isso mostra que quanto maior o porte das instâncias, mais adequada é a utilização das metaheurísticas propostas. Os métodos também alcançam um percentual considerável de melhores soluções (vitórias) no total de execuções realizadas. A BT apresentou melhor desempenho tanto na média dos resultados quanto no número de vitórias em ambos os conjuntos. Todavia, SA conseguiu um desempenho considerável com destaque nas instâncias de maior complexidade.

Agradecimentos

O presente trabalho contou com uma bolsa, processo: 107642/2009-3, do programa de iniciação científica PIBIC/CNPq da Universidade Federal de Lavras.

Referências

Akartunali, K., Miller, A.J.,(2009), A heuristic approach for big bucket multi-level production planning problems, *European Journal of Operational Research*, 193 (2), 396-411.

Almeder, C.,(2010), A hybrid optimization approach for multi-level capacitated lot-sizing problems, *European Journal of Operational Research*, 200, 599-606.
Glover, F., Laguna, M., Tabu Search, Kluwer, Norwell, MA, 1997.
Han, Y., Tang, J., Kaku, I., Mu, L.,(2009), Solving uncapacitated multilevel lot-sizing problems using a particle swarm optimization with flexible inertial weight, *Computers and Mathematics with Applications*, 57, 1748-1755.
Helber, S., Sahling, F.,(2010), A fix-and-optimize approach for the multi-level capacitated lot sizing problem, *International Journal of Production Economics*, 123, 247-256.
Jans, R., Degraeve, Z.,(2007), Metaheuristics for dynamic lot sizing: A review and comparison of solution approaches, *European Journal of Operational Research*, 177, 1855-1875.
Kirkpatrick, S., Gelatt, C. D., Vecchi, M.P.,(1983), Optimization by Simulated Annealing, Science, 220 (4598), 671-680.
Maes, J., McClain, J., van Wassenhove, N.,(1991), Multilevel capacitated lotsizing complexity and LP-based heuristics, *European Journal of Operational Research*, 53 (2), 131-148.
Özdamar, L., Barbarosoglu, G.,(2000), An integrated lagrangean relaxation-simulated annealing approach to the multi-level mulit-item capacitated lot sizing problem, *International Journal of Production Economics*, 68 (3), 319-331.
Sahling, F., Buschkühl, L., Tempelmeier, H., 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*, 37, 2546-2553.
Stadtler, H.,(1996), Mixed integer programming model formulations for dynamic multi-item multi-level capacitated lotsizing, *European Journal of Operational Research*, 94, 561-581.
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.
Tempelmeier, H., Derstroff,M.,(1996), A Lagrangian-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times, *Management Science*, 42 (5), 738-757.
Tempelmeier, H., Helber, S.,(1994), A heuristic for dynamic multi-item multi-level capacitated lotsizing for general product structures, *European Journal of Operational Research*, 75, 296-311.
Toledo, C.F.M., Oliveira, L., Oliveira, R.R.R., Ferreira, D.,(2009), Algoritmo Genético e Programação Matemática na Resolução de um Modelo Matemático para um Problema de Programaçao da Produçao de Bebidas, In: *XLI Simpósio Brasileiro de Pesquisa Operacional*, 2009, Porto Seguro, Brasil, 710-721.