ALGORITMOS BASEADOS EM SELEÇÃO CLONAL APLICADOS AO PROBLEMA GERAL DE DIMENSIONAMENTO DE LOTES E PROGRAMAÇÃO DA PRODUÇÃO

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

Michel Henrique Aquino Santos
Departamento de Ciência da Computação – Universidade Federal de Lavras
Campus Universitário, C.P. 3037, CEP 37200-000, Lavras, MG
michel.has@gmail.com

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

Márcio da Silva Arantes
Departamento de Ciência da Computação – Universidade Federal de Lavras
Campus Universitário, C.P. 3037, CEP 37200-000, Lavras, MG
marcio@comp.ufla.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 algoritmos baseados em seleção clonal para solucionar o Problema Geral de Dimensionamento de Lotes e Programação da Produção (PGDLPP) com e sem máquinas paralelas. Um modelo matemático é apresentado para este caso do PGDLPP assim como pseudocódigos para os métodos propostos. Instâncias do problema são solucionadas por uma ferramenta de modelagem matemática cujas soluções servem para avaliação do desempenho dos métodos apresentados. Os resultados obtidos também são comparados a um algoritmo genético com estrutura hierárquica de indivíduos, anteriormente utilizado para resolver o mesmo problema. Os resultados revelam um desempenho competitivo dos métodos de seleção clonal propostos.

PALAVRAS CHAVE. Programação da produção. Dimensionamento de lotes. Metaheurística. Área de classificação principal: MH Metaheurísticas

ABSTRACT

The present paper proposes clonal selection algorithms to solve the General Lot sizing and Scheduling Problem (GLSP) with and without parallel machines. A mathematical model is presented for this GLSP case as well as pseudo codes for the proposed methods. Problem instances are solved by mathematical modeling computational package whose solutions will be benchmarks to evaluate the performance of the presented methods. The results found are also compared with a hierarchical structure genetic algorithm previously used to solve the same problem. The results report a competitive performance for the clonal selection methods proposed.

KEYWORDS. Scheduling. Lot sizing. Metaheuristic. Main area: Metaheuristics.

1. Introdução

O presente trabalho propõe uma metaheurística, baseada no sistema imunológico biológico e conhecida como algoritmo de seleção clonal (CLONALG), para solucionar o Problema Geral de Dimensionamento de Lotes e Programação da Produção (PGDLPP) com máquinas simples e paralelas. O PGDLPP estudado também considera penalização para demandas não atendidas, onde uma formulação matemática é apresentada para essa variante do problema. Instâncias são geradas para o PGDLPP estudado seguindo os critérios propostos por Haase (1996) para o mesmo problema. Todas as instâncias geradas são solucionadas usando a ferramenta de modelagem matemática GAMS/CPLEX (GAMS, 2009). As soluções obtidas servem como ponto de partida na avaliação de desempenho do método proposto. Além disso, os resultados são comparados aqueles encontrados pelo Algoritmo Genético com estrutura hierárquica de indivíduos proposto em Toledo *et al* (2009) para o mesmo problema.

Uma revisão sobre formulações matemáticas e métodos de resoluções para variações de problemas de dimensionamento de lotes e programação da produção pode ser encontrada em Drexl e Kimms (1997) e Karimi *et al* (2003). Estudos considerando o PGDLPP com máquinas paralelas são descritos em Kang *et al.* (1999) e Meyr (2002). O primeiro trabalho trata o PGDLPP com máquinas paralelas usando um método baseado em geração de coluna e *branch & bound*. O segundo utiliza a metaheurística *simulated annealing* para determinar as variáveis binárias (atribuição de produtos às linhas e períodos) do PGDLPP e um algoritmo exato para determinar as variáveis contínuas (dimensionamento de lotes e definição dos estoques). O dimensionamento de lote e a programação da produção com restrição de capacidade é um problema de otimização NP–Difícil (Bitran e Yanasse, 1982). O problema de dimensionamento de lotes e programação da produção multi-item também é um problema NP-Difícil (Chen e Thizy, 1990). Luche *et al.* (2009) e Clark *et al.* (2009) apresentam modelos de programação inteiro-misto para esse tipo de problema no contexto da produção de grãos eletrofundidos e de ração animal. A complexidade desses problemas faz com que sejam propostas heurísticas ou metaheurísticas como métodos de resolução. Jans e Degraeve (2007) revisam a literatura considerando o uso de metaheurística em problemas de dimensionamento de lote e programação da produção.

Um sistema imunológico artificial é desenvolvido como uma metáfora do sistema imunológico natural (Timmis, 2000). Dasgupta (1998) também define sistemas imunológicos artificiais como sendo formados por metodologias inteligentes inspiradas no sistema imunológico biológico. O algoritmo de seleção clonal (CLONALG) foi proposto por De Castro e Von Zuben (2001). Trata-se de um algoritmo que utiliza o princípio da seleção clonal, onde células capazes de reconhecer os invasores (antígenos) proliferam através da geração de um grande número de clones. Durante essa proliferação, os clones passam por um processo de mutação com o objetivo de atingir maior afinidade com o antígeno a ser combatido.

Um Algoritmo Genético (AG) foi utilizado por Toledo *et al* (2009) para solucionar o PGDLPP aqui apresentado. Os autores utilizaram uma representação em árvore ternária para os indivíduos, onde indivíduos são separados em cluster com uma ordem de hierarquia entre eles. O AG proposto foi capaz de encontrar boas soluções, obtendo melhor desempenho em instâncias do PGDLPP com máquinas paralelas. Algoritmos Genéticos (AG) são métodos de computação evolutiva que simulam processos biológicos (Holland, 1975). A próxima seção descreve o modelo matemático para o PGDLPP estudado neste trabalho. A Seção 3 apresenta o algoritmo CLONALG proposto. Os resultados computacionais obtidos são descritos na Seção 4. As conclusões do trabalho são apresentadas na Seção 5.

2. Modelo matemático para o PGDLPP

Um modelo matemático será apresentado para o PGDLPP, conforme proposto em Toledo *et al* (2009), considerando máquinas paralelas e penalização das demandas não atendidas.

A formulação apresentada segue a modelagem proposta por Meyr (2002) para o PGDLPP com máquinas paralelas. A principal diferença está na inserção de variáveis que acumulam as demandas não atendidas para cada produto. Essas variáveis são incluídas na equação de balanço de estoque no primeiro período de produção e também são penalizadas na função objetivo do modelo. O problema é modelado dividindo o horizonte de planejamento em T macro-períodos. Por sua vez, cada macro-período t possui um número fixo de micro-períodos S. Um aspecto interessante neste tipo de modelagem é que o tamanho de cada micro-período s varia de forma proporcional ao tamanho do lote do produto a ele atribuído. Assim, as variáveis de dimensionamento dos lotes e de atribuição de produtos às linhas e períodos estão indexadas por produtos e micro-períodos. O modelo assume que um único produto é atribuído e produzido em cada micro-período. Um total de J produtos e L linhas são considerados. Abaixo são listados os demais parâmetros do problema

- C_t: Capacidade em unidades de tempo disponível no macro-período t.
- $TP_{l,j}$: Tempo de processamento do produto j na linha l.
- Min_j: Lote mínimo do produto j.
- H_j: Custo de estoque do produto j.
- CT_{ij}: Custo de troca do produto i para o produto j.
- D_{it}: Demanda do produto i no macro-período t.
- I_{j0}: Estoque inicial do produto j no início do horizonte de tempo.
- $Y_{l,j,0}$: 1, se o produto j está ajustado inicialmente para a linha l; 0 caso contrário.
- M: Penalização por unidade de demanda não atendida.

As variáveis do modelo são apresentadas a seguir:

- $I_{jt} \geq 0$: Estoque do produto j ao final do macro-período t.
- $q_{ljs} \geq 0$: Quantidade do produto j produzido no micro-período s da linha l.
- $q^0_j \geq 0$: Quantidade de demanda do produto j que não foi produzida.
- $y_{l,j,s}$: 1 Se o produto j é atribuído à linha l no micro-período s; 0 caso contrário.
- $z_{lijs} \geq 0$: $z_{lijs} = 1$ se há troca do produto i para j no micro-período s da linha l; $z_{ijs} = 0$, caso contrário.

Modelo matemático para o PGLDPP com máquinas paralelas e penalização de demandas:

$$Minimize \sum_{j=1}^{J}\sum_{t=1}^{T} H_j I_{jt} + \sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T.S} CT_{ij} Z_{lijs} + M\sum_{j=1}^{J} q_j^0 \qquad (1)$$

$$I_{j1} = I_{j,0} + q_j^0 + \sum_{l=1}^{L}\sum_{s=1}^{S} q_{ljs} - D_{j1} \qquad j = 1, \ldots, J \qquad (2a)$$

$$I_{jt} = I_{jt-1} + \sum_{l=1}^{L}\sum_{s=(t-1)S+1}^{tS} q_{ljs} + q_j^0 - D_{jt} \qquad j = 1, \ldots, J,\ t=1, \ldots, T \qquad (2b)$$

$$\sum_{j=1}^{J}\sum_{s=(t-1)S+1}^{tS} TP_{lj} q_{ljs} \leq C_t \qquad l = 1, \ldots, L,\ t=1, \ldots, T \qquad (3)$$

$$q_{ljs} \leq \frac{C_t}{TP_{lj}} y_{ljs} \qquad l = 1, \ldots, L,\ j=1, \ldots, J,\ s=1, \ldots, T.S \qquad (4)$$

$$q_{ljs} \geq Min_j \left(y_{ljs} - y_{lj,s-1}\right) \qquad l = 1, \ldots, L,\ j=1, \ldots, J,\ s=1, \ldots, T.S \qquad (5)$$

$$\sum_{j=1}^{J} y_{ljs} = 1 \qquad l = 1, \ldots, L,\ s=1, \ldots, T.S \qquad (6)$$

$$z_{ijs} \geq y_{i,s-1} + y_{j,s} - 1 \qquad i = 1, \ldots, L_t \; i=1, \ldots, J, \; j=1, \ldots, J, \; s=1, \ldots, T.S \quad (7)$$

A função objetivo minimiza os custos de estoque e os custos de troca envolvendo produtos. Os custos de troca foram considerados como dependentes da sequência dos produtos, mas independentes das linhas. As demandas não atendidas são acumuladas nas variáveis q^0_j e penalizadas na função objetivo. O estoque de cada macro-período é determinado pelas equações (2a) e (2b). Observe que a demanda não atendida é acumulada na equação de balanço de estoque do primeiro período (2a). Isso permite que as demandas não atendidas nos demais períodos continuem satisfazendo a restrição de balanço de estoque já que são acumuladas em q^0_j. Considera-se que essas demandas foram produzidas em um período t=0 com capacidade ilimitada. Esse tipo de representação foi utilizado por Toledo *et al* (2007) num modelo integrado de dimensionamento de lotes e programação da produção em fábricas de bebidas. A restrição (3) garante que a capacidade disponível dentro do período t não seja violada. A capacidade disponível por macro-período é a mesma para todas as linhas, mas cada linha pode ter diferentes tempos de processamento por produto ($TP_{l,j}$). O tamanho máximo do lote de um produto em um micro-período é dado pela restrição (4). Essa restrição também impõe que nada seja produzido, caso não ocorra atribuição do produto ao micro-período ($y_{l,j,s}$ = 0). A restrição (5) assegura que um lote mínimo será produzido quando ocorre efetivamente a atribuição de um novo produto a um micro-período ($y_{l,j,s}$ = 1). Por outro lado, se o mesmo produto é atribuído a dois micro-períodos consecutivos, não necessariamente haverá produção na segunda atribuição. A equação (6) permite que somente um produto seja produzido em um micro-período. As trocas de produtos são determinadas pela restrição (7).

3. Algoritmo de seleção clonal proposto

Inicialmente será descrita a forma como a solução é representada no método de seleção clonal (CLONALG) apresentado. Em seguida, os pseudocódigos dos métodos são detalhados.

3.1 Representação e avaliação das soluções

A representação utilizada trata-se de uma extensão para máquinas paralelas da representação proposta em Fleischmann e Meyr (1997) para o PGDLPP com uma única máquina. A representação da solução consiste em uma lista de máquinas ou linhas, onde cada linha armazena informações sobre sua sequência de produção. Duas posições consecutivas em uma sequência de produção não podem ser ocupadas pelo mesmo produto. A única exceção ocorre para a última posição de um macro-período e a primeira posição do macro-período seguinte que podem ser idênticas. Também não pode haver uma posição sem produto entre duas posições ocupadas em um mesmo macro-período. Suponha uma instância do problema com duas linhas, três produtos e dois macro-períodos divididos em três micro-períodos cada.

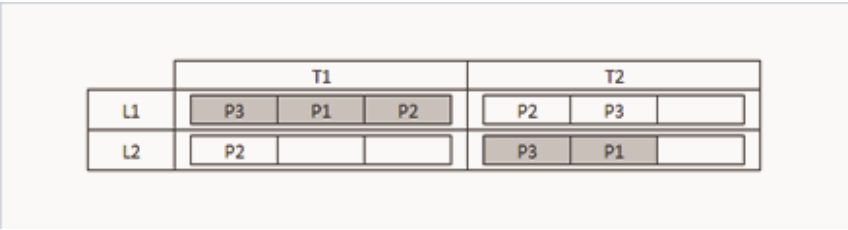

Figura 1– Exemplo de representação da solução.

A Figura 1 exemplifica a representação de uma possível solução. Observe que a linha *L1* possui três produtos alocados no macro-período *T1* e dois alocados em *T2*. A ordem dos produtos em cada período representa a sequência em que serão produzidos. Na linha *L2*, um único produto será produzido em *T1* e três produtos em *T2*. A avaliação de uma solução consiste no cálculo do valor da função objetivo (equação 1) apresentada na seção 2. Esse processo exige a determinação dos custos de troca, estoque e a penalização por demanda não atendida. Os custos de troca são

calculados a partir da sequência de produtos existentes em cada linha na representação da solução. Os custos de estoque e a penalidade por demanda não atendida exigem a determinação do dimensionamento dos lotes a partir das informações codificadas na representação da solução. Esse dimensionamento é feito através do método apresentado no pseudocódigo a seguir (Figura 2).

```
Método de dimensionamento de lotes
   t=T
   Enquanto (t≥1) faça
      //-----Garantindo lote mínimo------
      Para toda linha l
         Para todo produto com (Yljt>0)
            p = Mínimo( Minj * Yljt ; Dj - ACjt );                    //Calcula a
parcela a ser produzida
            Atualiza Xljt, Klt, Dj;
            Se Klt < 0 então Penalizar capacidade violada

      //-----Evitando infactibilidades--------------
      Para todo produto j
         Q = Dj - LSjt;
         Para toda linha l com (Yljt>0 e Q > 0)
            p = Mínimo( Dj - ACjt ; Klt/TPlj ; Q );
            Atualiza Xljt, Klt, Dj;

      //------Distribui em uma ordem gulosa-----
      Para toda linha l e produto j  com (Yljt > 0 ) em ordem decrecente de
Hj/TPlj
         p = Mínimo( Dj - ACjt ; Klt/TPlj );
         Atualiza Xljt, Klt, Dj;
   t = t-1
   fim Enquanto

   Qj = Dj, para todo j.
   //----Distribui as demandas dos macro-período para os micro-períodos---
--
   Para todo l,j,t e s pertencente a St
      Se Yljt > 0 então
         Xljs = Xljt / Yljt;
      Senão
         Xljs = 0;
fim do método.
```

Figura 2 – Pseudocódigo para o dimensionamento dos lotes

Esse método é baseado no algoritmo *greedy-mod* proposto em Fleischmann & Meyr (1997) para o PGDLPP com máquinas simples. O método de decodificação é determinístico e procura estabelecer o tamanho dos lotes, minimizando o custo de estoque. Os parâmetros do algoritmo são descritos abaixo:

- Xljt - armazena o lote do produto j produzido na linha l no macro-período t.
- Yljt - número de lotes do produto j no macro-período t na linha l.
- Dj – demanda acumulada (total) do produto j.

- ACjt - demanda acumulada do produto j até o macro-periodo anterior ao t.
- LSjt - limite superior estimado considerando que toda a capacidade disponível fosse utilizada para produzir o produto j em períodos anteriores ao t.

$$LS_{jt} = 0; \text{ para t=1 e } LS_{jt} = LS_{j(t-1)} + \sum_{l=1}^{L} \frac{Minimo(1, Y_{l,jt}) * B_{lt}}{TP_{lj}} \text{ ; para t>1}$$

- Xljs – lote do produto j produzido na linha l no micro-período s.
- Qj – quantia de demanda do produto j não alocada pelo metodo de decodificação.

O algoritmo trabalha preenchendo as quantias dos lotes em ordem decrescente de macro-período. Primeiro aloca o lote mínimo para os produtos atribuídos a cada micro-período na sequência de produção das linhas. Em seguida, dimensiona os lotes para produtos que, se não forem produzidos no macro-período corrente, não haverá capacidade disponível para sua produção, ou o produto não aparecerá em outra posição na matriz de sequência das linhas. O objetivo é evitar infactibilidades. Por último, todos os produtos alocados na matriz de sequência das linhas têm seus lotes redimensionados, seguindo a ordem decrescente do quociente Hj/TPlj. A Figura 3 apresenta o procedimento que atualiza as quantias Xljt, Klt e Dj pela parcela *p* da demanda que será produzida.

```
Atualiza Xljt, Klt, Dj dada uma parcela de demanda p calculada
        Xljt = Xljt + p;          //Adiciona a parcela, a quantia a ser
produzida
        Klt = Klt - p * TPlj;     //Subtrai a capacidade para produzir a
quantia p.
        Dj = Dj - p;              //Nova demanda do produto que deverá ser
atendida.
```

Figura 3 – Atualizar Xljt, Klt, Dj

3.2 Algoritmo de seleção clonal (CLONALG)

Duas abordagens para o algoritmo de seleção clonal serão propostas baseadas nas idéias apresentadas por De Castro e Von Zuben (2001). A Figura 4 apresenta o pseudocódigo para o primeiro algoritmo de seleção clonal chamado ASC-1.

```
Procedimento ASC-1
  Inicializa(vetorAnticorpos);
  CalculaFitness(vetorAnticorpos);
  Ordena(vetoAnticorpos);
  Enquanto (tempo total não atingido) faça
     Para a ← 1 até nAnticorpos faça
        Para c ← 1 até (taxaClonagem*nAnticorpos/a) faça
              cloneNum← cloneNum + 1;
              vetorClones[cloneNum] ← vetorAnticorpos[a];
              Mutar(vetorClones[cloneNum]);
              CalculaFitness(vetorClones[cloneNum]);
        Fim Para;
     Fim Para;
     Ordena(vetorClones);
     Insere(vetorClones,vetorAnticorpos);
     Ordena(vetorAnticorpos);
     ReiniciaPiores(vetorAnticorpos);
     Ordena(vetorAnticorpos);
```

```
  Fim Enquanto;
Fim ASC-1;
```

Figura 4 – Pseudocódigo ASC-1.

A inicialização dos anticorpos consiste na geração de diversas representações da solução. Essa geração ocorre alocando de forma aleatória produtos na sequência existente em cada linha. Na avaliação da população de anticorpos (*CalculaFitness* (*vetorAnticorpos*)), o método de decodificação da solução é executado e a função de avaliação é calculada. A função de avaliação é a mesma apresentada na equação 1 da seção 2. No PGDLPP, o melhor anticorpo é aquele que tem o menor valor retornado pela função de avaliação. A ordenação dos anticorpos (*ordena(vetorAnticorpos)*) pode determinar três grupos distintos. O primeiro grupo forma a memória que é composta pelos anticorpos com melhor valor na função de avaliação. O segundo grupo consiste nos anticorpos intermediários e o último grupo contém os piores anticorpos.

O ASC-1 clona todos os anticorpos com taxa proporcional a sua afinidade. Esses clones sofrem mutação de forma inversamente proporcional a sua afinidade. Em seguida, todos os clones gerados são avaliados. Ao final do processo de clonagem e mutação dos anticorpos, os clones gerados são ordenados. Em seguida, os clones com melhor valor de função de avaliação são inseridos na memória (*Insere*(*vetorClones*,*vetorAnticorpos*);). Isso ocorre desde que o valor da função de avaliação seja melhor que o de algum anticorpo da memória. Além disso, o clone a ser inserido não pode ser idêntico a um anticorpo da população. Os piores anticorpos da população irão ser reinicializados (*reiniciaPiores(vetorAnticorpos))* e a população é novamente ordenada. A Figura 5 apresenta o pseudocódigo para a segunda abordagem chamada ASC-2.

```
Procedimento ASC-2
  Inicializa(vetorAnticorpos);
  CalculaFitness(vetorAnticorpos);
  Ordena(vetoAnticorpos);
  Enquanto (tempo total não atingido) faça
     Para a ← 1 até nAnticorpos faça
        Para c ← 1 até (taxaClonagem*nAnticorpos/a) faça
              cloneNum← cloneNum + 1;
              vetorClones[cloneNum] ← vetorAnticorpos[a];
              Mutar(vetorClones[cloneNum]);
              CalculaFitness(vetorClones[cloneNum]);
        Fim Para;
        Ordena(vetorClones);
        Insere(vetorClones,vetorAnticorpos[a]);
     Fim Para;
     Ordena(vetorAnticorpos);
     ReiniciaPiores(vetorAnticorpos);
     Ordena(vetorAnticorpos);
  Fim Enquanto;
Fim ASC-2;
```

Figura 5 – Pseudocódigo ASC-2.

Na segunda abordagem, o processo de inserção dos clones é modificado. O grupo de clones gerados para um anticorpo é avaliado pelo método *CalculaFitness()* e ordenados em *Ordena*(*vetorClones*). Os anticorpos são substituídos pelo clone de melhor valor gerado a partir deles. No anticorpo da memória, essa substituição ocorrerá apenas se o melhor clone superar o valor do anticorpo da memória. Os demais anticorpos serão substituídos sempre pelo melhor clone gerado a partir deles. Ao final, todos os anticorpos são ordenados e os piores são reinicializados. No método *Mutar* (), a taxa de mutação define qual é a quantidade de mutações a ser executada considerando o total de posições existentes na representação da solução. Por exemplo, uma instância com duas linhas, dois períodos e quatro micro-períodos terá um total de

posições existentes na representação da solução de 2x2x4=16 (linha x macro-período x micro-período). Se a taxa de mutação utilizada for de 20%, um total de 3 mutações serão executadas. O tipo de mutação a ser executada será aleatoriamente selecionado entre os oito tipos definidos neste trabalho:

1. Insere um Lote: Cria um novo lote e o insere;
2. Deleta um Lote: Sorteia um micro-período e exclui o lote;
3. Troca Lote Micro: Troca dois lotes dentro de uma linha e um macro-período.
4. Troca Lote Linha: Troca dois lotes em linhas diferentes de um mesmo macro-período, situados em micro-períodos aleatoriamente selecionados.
5. Troca Lote Macro: Troca dois lotes em macros diferentes, considerando uma linha e micro-período aleatoriamente selecionados.
6. Troca Sequência Produção Macro: Sorteia duas linhas e um macro-período, trocando a sequência de produção
7. Troca Sequência Producao Linha: Sorteia uma linha, dois macro-períodos e troca a sequência de produção
8. Troca Sequência Producao Linha Macro: Sorteia duas linhas, dois macros-períodos e troca as sequências de produção

4. Resultados Computacionais

Os testes computacionais foram executados em instâncias do PGDLPP com máquina simples e máquinas paralelas. As instâncias são as mesmas definidas e solucionadas em Toledo *et al* (2009). No PGDLPP com máquinas simples, os autores utilizaram um total de 4 conjuntos com 10 instâncias cada com os parâmetros apresentados na Tabela 1. Os valores dos parâmetros foram estabelecidos por Haase (1996) na resolução do PGDLPP com máquinas simples, onde U(%) é a porcentagem de utilização da capacidade disponível. Todos os demais parâmetros do problema também foram ajustados pelos mesmos valores utilizados por Haase (1996): $d_{jt} \in \{0,1,...,100\}$, $H_j=1, TP_{1j}=1$, $CT_{ij} \in \{100,...,200\}$ para $i \neq j$ e $CT_{ij}=0$ para $i=j$, $M=10000$ e $Min_j = 1$. O modelo matemático da seção 2 e as instâncias geradas foram codificados utilizando o pacote computacional GAMS e solucionados pelo solver CPLEX 12.0 com tempo de execução de 1 hora.

Tabela 1 – Conjuntos de Instâncias para o PGDLPP com uma única máquina.

Conjuntos	**Macro-Períodos (*T*)**	**Micro-Períodos (*S*)**	**Produtos (*J*)**	**Capacidade (C_t)**	***U*(%)**
S1	6	4	4	200	80
S2	6	4	4	200	90
S3	5	4	4	200	90
S4	5	5	5	250	90

Os resultados obtidos pelos algoritmos de seleção clonal ASC-1 e ASC-2 são comparados aos retornados pelo GAMS/CPLEX e pelo Algoritmo Genético (AG) também apresentados em Toledo *et al* (2009). O AG foi executado utilizando uma população com estrutura hierárquica do tipo ternária, 40 indivíduos, 2,0 de taxa de *crossover* do tipo 1 ponto e 0,7 de taxa de mutação. ASC-1 foi executada com 100 anticorpos, sendo 30 na memória, 40 na população intermediária e 30 nos piores. Também foi utilizada taxa de clonagem de 0,5. ASC-2 foi executada com 100 anticorpos, sendo 50 na memória, 30 na população intermediária e 20 piores. A mesma taxa de clonagem foi utilizada. Todos os testes foram realizados em um processador Intel Core 2 Duo, 2,66 GHz e 2 GB RAM. As metaheurísticas executaram 10 vezes em cada uma das 10 instâncias de cada conjunto. O tempo de execução por instância foi limitado a 30 minutos, ou seja, metade do tempo dado ao método exato. As Tabelas 2 e 3 apresentam os resultados para os quatro conjuntos de instâncias do PGDLPP com máquinas simples.

Tabela 2 – Resultados obtidos para o PGDLPP com uma única máquina em S1 e S2.

S1	GAMS	GA	ASC-1	ASC-2	S2	GAMS	GA	ASC-1	ASC-2
S1-0	1658*	1658	1658	1658	**S2-0**	1729*	1731,8	1742,6	1730,3
		0,0	0,0	0,0			0,2	0,8	0,1
S1-1	1421*	1421	1421	1421	**S2-1**	1758*	1758	1798	1758
		0,0	0,0	0,0			0,0	2,3	0,0
S1-2	1700*	1703	1703	1704,8	**S2-2**	2162*	2162	2195,7	2162
		0,2	0,2	0,3			0,0	1,6	0,0
S1-3	1459*	1471	1471	1471	**S2-3**	1747*	1753,8	1771,3	1747
		0,8	0,8	0,8			0,4	1,4	0,0
S1-4	1543*	1551,4	1548,6	1543	**S2-4**	1638*	1638	1638	1638
		0,5	0,4	0,0			0,0	0,0	0,0
S1-5	1402*	1402	1402	1402	**S2-5**	1534*	1539	1539	1539
		0,0	0,0	0,0			0,3	0,3	0,3
S1-6	1477*	1482,6	1493	1480	**S2-6**	1533*	1533	1533,0	1533
		0,4	1,1	0,2			0,0	0,0	0,0
S1-7	1463*	1463	1463	1463	**S2-7**	1822*	1831	1824,8	1822
		0,0	0,0	0,0			0,5	0,2	0,0
S1-8	1480*	1487	1487	1487	**S2-8**	1863*	1863	1863	1863
		0,5	0,5	0,5			0,0	0,0	0,0
S1-9	1626*	1630,7	1633	1628,6	**S2-9**	1751*	1751	1764,1	1751
		0,3	0,4	0,2			0,0	0,7	0,0

Tabela 3 – Resultados obtidos para o PGDLPP com uma única máquina em S3 e S4.

S3	GAMS	GA	ASC-1	ASC-2	S4	GAMS	GA	ASC-1	ASC-2
S3-0	1269*	1279	1279	1279	**S4-0**	1833	1840,8	1853,9	1833
		0,8	0,8	0,8			0,4	1,1	0,0
S3-1	1491*	1490	1493,6	1491	**S4-1**	1986	1989,6	2006,6	1986
		-0,1	0,2	0,0			0,2	1,0	0,0
S3-2	1338*	1338	1338	1338	**S4-2**	1825	1821,8	1832,7	1805
		0,0	0,0	0,0			-0,2	0,4	-1,1
S3-3	1530*	1530	1530	1530	**S4-3**	1674	1674	1694,9	1678
		0,0	0,0	0,0			0,0	1,2	0,2
S3-4	1663*	1663,8	1664,3	1663	**S4-4**	1974	1974	1994,5	1974
		0,0	0,1	0,0			0,0	1,0	0,0
S3-5	1542*	1542	1542	1542	**S4-5**	1994	2010,9	2012,4	1994
		0,0	0,0	0,0			0,8	0,9	0,0
S3-6	1553*	1553	1553	1553	**S4-6**	1790	1790	1814,6	1791,2
		0,0	0,0	0,0			0,0	1,4	0,1
S3-7	1518*	1533,6	1538,8	1518	**S4-7**	1786	1784,8	1781,1	1777
		1,0	1,4	0,0			-0,1	-0,3	-0,5
S3-8	1526*	1526	1526	1526	**S4-8**	2150	2139	2148,2	2139
		0,0	0,0	0,0			-0,5	-0,1	-0,5
S3-9	1441*	1441	1441	1441	**S4-9**	1782	1782,8	1782,0	1782
		0,0	0,0	0,0			0,0	0,0	0,0

Nas Tabelas 2 e 3, são listados os resultados obtidos pelo GAMS/CPLEX e o valor médio considerando a solução final obtida nas 10 execuções das metaheurísticas. Também é apresentado o desvio $Desv(\%)=100*(Z-Z^*)/Z^*$, onde Z é o valor médio da solução final obtida pela metaheurística e Z^* á o valor da solução final retornada pelo GAMS/CPLEX. Os valores na coluna GAMS com * indicam solução ótima. O GAMS/CPLEX retornou a solução ótima ao final

de todas as execuções em S1, S2 e S3. No conjunto S4, o GAMS/CPLEX retornou a melhor solução factível encontrada após 1 hora de execução. As metaheurísticas ASC-1, ASC-2 e AG foram capazes de encontrar a solução ótima na maioria das instâncias em S1, S2 e S3 com um desempenho bastante aproximado em cada instância. A Figura 6 apresenta os desvios médios considerando todo o conjunto de instâncias. Todos os métodos obtiveram desvios abaixo de 1% com ASC-2 obtendo o menor desvio médio em todos os conjuntos de instâncias. No caso do conjunto S4, onde soluções ótimas não foram encontradas pelo GAMS/CPLEX, ASC-2 conseguiu inclusive melhorar em média as soluções factíveis obtidas pelo *solver*.

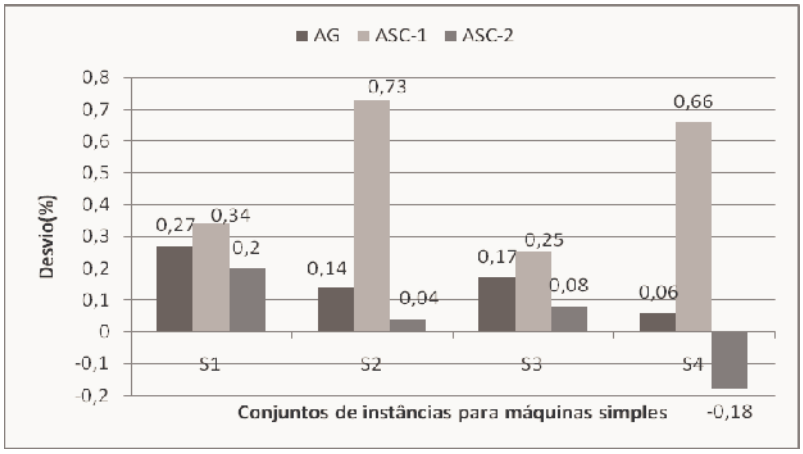

Figura 6 – Desvios médios nas instâncias S1,S2,S3 e S4.

O PGDLPP com máquinas paralelas foi solucionado usando os mesmos três conjuntos de instâncias (P1, P2 e P3) apresentados em Toledo *et al* (2009). Os parâmetros utilizados na geração dessas instâncias foram T=6, S=4, J=4, C_t=200, onde apenas o número de linhas variou com L=2 em P1, L=3 em P2 e L=4 em P4. Os demais parâmetros permaneceram como antes, exceto TP_{lj} e U. Agora $TP_{lj} \in [0,1]$ passa a variar por produto e linha e a taxa de utilização foi fixada em U= 80%. Um total de 5 instâncias foi gerado para cada um desses conjuntos. O GAMS/CPLEX também foi utilizado na resolução exata dessas instâncias com tempo de execução de 1 hora. As metaheurísticas foram executadas por 30 minutos. Assim como nas máquinas simples, tiveram metade do tempo destinado ao método exato. As Tabela 4 e 5 apresentam os resultados obtidos.

Tabela 4 – Resultados obtidos para o PGDLPP com máquinas paralelas em P1 e P2.

P1	GAMS	GA	ASC-1	ASC-2	P2	GAMS	GA	ASC-1	ASC-2
P1-0	5819,8	4291,4	4481,5	4290,9	**P2-0**	5730,2	3803,5	3892,2	3800,6
		-26,3	-23,0	-26,3			-33,6	-32,1	-33,7
P1-1	5561	4221,4	4334,4	4243,5	**P2-1**	5364,3	3844,2	3954,6	3892,1
		-24,1	-22,1	-23,7			-28,3	-26,3	-27,4
P1-2	4371,5	3742,8	3877,8	3666,2	**P2-2**	6375,3	4109,8	4418,8	4283,4
		-14,4	-11,3	-16,1			-35,5	-30,7	-32,8
P1-3	5043,8	4017,8	4235,3	4038,0	**P2-3**	7045,4	4101,6	4263,7	4104,7
		-20,3	-16,0	-19,9			-41,8	-39,5	-41,7
P1-4	4536,9	4113,8	4259,8	4192,2	**P2-4**	6666,0	4056,0	4220,2	4144,9
		-9,3	-6,1	-7,6			-39,2	-36,7	-37,8

Tabela 5 – Resultados obtidos para o PGDLPP com máquinas paralelas em P3.

P3	GAMS	GA	ASC-1	ASC-2
P3-0	5679,6	3733,4	3620,5	3761,0
		-34,3	-36,3	-33,8

P3-1	7713,1	3699,0	3705,4	3677,1
		-52,0	-52,0	-52,3
P3-2	7806,1	3607,5	3540,7	3647,8
		-53,8	-54,6	-53,3
P3-4	7836,7	3682,9	3587,6	3800,1
		-53,0	-54,2	-51,5
P3-5	7130,8	3529,5	3463,2	3546,7
		-50,5	-51,4	-50,3

O GAMS/CPLEX não foi capaz de retornar soluções ótimas ao final de 1 hora de execução em P1, P2 e P3. Por outro lado, todas as metaheurísticas superaram os valores das soluções finais obtidas pelo GAMS/CPLEX em todas as instâncias. ASC-2 conseguiu 3 melhores resultados em P1, enquanto AG obteve 4 melhores resultados em P2 e ASC- 1 ficou com 4 melhores resultados em P3. Uma comparação entre as médias dos desvios por conjunto de instâncias é apresentada na Figura 7. AG e ASC-2 apresentam desvios bastante próximos nos três conjuntos de instâncias propostos para o PGDLPP com máquinas paralelas. ASC-1 obteve médias inferiores em P1 e P2, mas alcançou melhores resultados em P3. Os três métodos não diferem de forma considerável e as abordagens propostas se mostram competitivas no PGDLPP com máquinas paralelas.

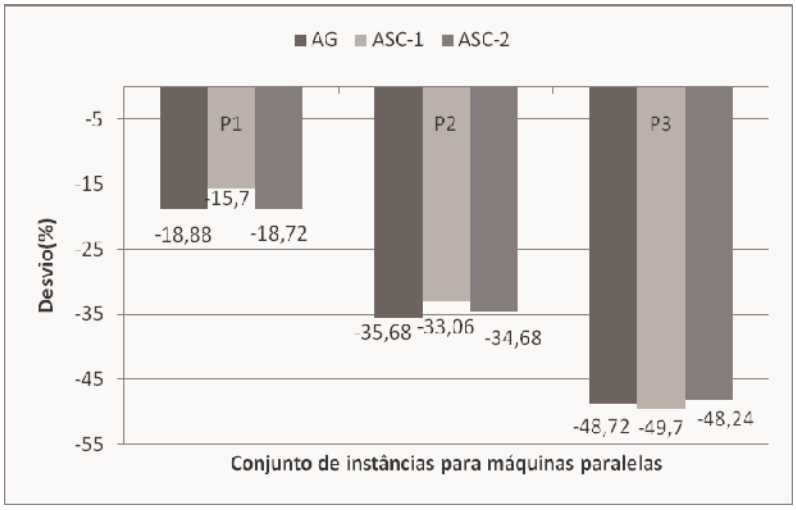

Figura 7 – Desvios médios nas instâncias P1,P2 e P3.

4. Conclusões

O presente trabalho propôs métodos de resolução para o Problema Geral de Dimensionamento de Lotes e Programação da Produção (PGDLPP) com e sem máquinas em paralelo, e com penalização por demandas não atendidas. Um modelo matemático foi apresentado e duas abordagens baseadas no algoritmo de seleção clonal foram propostas. A primeira abordagem ASC1 utiliza um critério de inserção que prioriza a substituição dos melhores clones na memória, ou seja, na atualização dos melhores anticorpos. A segunda abordagem prioriza a substituição de todos os anticorpos pelos clones por ele gerados. No caso da dos anticorpos da memória, serão substituídos apenas se forem piores que seus clones. Um total de 7 conjuntos de instâncias são solucionados e comparados aos resultados obtidos pelo GAMS/CPLEX e por um algoritmo genético com estrutura hierárquica de indivíduos. Nas instâncias do PGDLPP com máquinas simples, tanto ASC-1 quanto ASC-2 apresentam desvios abaixo de 0.5% em relação ao GAMS/CPLEX. ASC-2 consegue o melhor desempenho em todos

os conjuntos de instâncias com máquinas simples. Os métodos superam ou igualam a maioria dos resultados obtidos pelo AG. Não há uma diferença relevante de desempenho médio entre ASC-1, ASC-2 e o AG nas instâncias do PGDLPP com máquinas paralelas. Há um destaque para ASC-1 que consegue desempenho superior no conjunto P3. Dessa forma, os métodos baseados em seleção clonal se mostram competitivos na resolução das instancias utilizadas para representar o PGDLPP. A inserção mais elitista dos clones se mostrou indicada nas instâncias do PGDLPP com máquinas simples. A substituição na população intermediária se mostrou mais eficiente na resolução do conjunto mais complexo de instâncias do PGDLPP com máquinas paralelas.

Agradecimentos

O presente trabalho contou com uma bolsa do programa de iniciação científica PIBIC/CNPq e outra bolsa do programa de iniciação científica PIBIC/FAPEMIG da Universidade Federal de Lavras.

References

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

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

Clark, A., Morabito, R. and Toso, E. (2009), Production setup-sequencing and lot-sizing at an animal nutrition plant through ATSP subtour elimination and patching, *Journal of Scheduling*, 10.1007/s10951-009-0135-7.

Dasgupta, D., (1998),*Artificial Immune Systems and Their Applications*, Springer-Verlag.

dE CASTRO, L.N. & VON ZUBEM, F.J., (2002), IEEE Learning and Optimization Using the Clonal Selection Principle, *IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems*, Vol. 6, no. 3, p. 239-251.

Drexl, A., Kimms, (1997), A. Lot sizing and scheduling – survey and extensions, *European Journal of Operational Research* 99, 221–235.

Fleischmann, B., Meyr, H., (1997), The general lotsizing and scheduling problem, *ORSpektrum*, 19, 11-21.

GAMS, (2010), General Algebraic Modeling System, *website*: http://www.gams.com. Acesso em 19 de maio de 2010.

Haase, K.(1996), Capacitated lot-sizing with sequence dependent setup costs, *ORSpektrum*, 8, 51-59.

Holland, J.H., *Adaptation in natural and artificial systems*, The University of Michigan Press, 1975.

Jans, R. e Degraeve, Z., (2007), Metaheuristics for dynamic lot sizing: A review and comparison of solution approaches, *European Journal of Operational Research*, 177, 1855-1875.

Kang, S., Malik, K., Thomas, L.J., (1999), Lotsizing and scheduling on parallel machines with sequence-dependent setup costs, *Management Science*, 45, 273-289.

Luche, J.R.D., Morabito, R., Pureza, V. (2009). Combining process selection and lot sizing models for production scheduling of electrofused grains. *Asia-Pacific Journal of Operational Research*, 26 (3): 421-443.

Meyr, H., (2002), Simultaneous lotsizing and scheduling on parallel machines, *European Journal of Operational Research*, 139, 277-292.

Timmis, J. *Artificial Immune Systems: A Novel Data Analysis Technique Inspired by the Immune Network Theory*, Tese de Doutorado, Departament of Computer Science, University of Whales, 2000.

Toledo, C.F.M., Ferreira, J. E., Simeone, F. & Rosa, G.P., (2009), Metaheurísticas Aplicadas ao Problema Geral de Dimensionamento de Lotes e Programação da Produção**.** *In: XLI SBPO Simpósio Brasileiro de Pesquisa Operacional*, Porto Seguro, BA.