UM ALGORITMO GRASP PARA SELEÇÃO DE PROCESSOS DE PRODUÇÃO NA PROGRAMAÇÃO DA PRODUÇÃO DE GRÃOS ELETROFUNDIDOS

José Roberto Dale Luche
Instituto Federal de São Paulo
Avenida Zélia de Lima Rosa, 100 – Boituva, SP – CEP: 18550-000
dluche@gmail.com

Vitória Pureza
Departamento de Engenharia de Produção, Universidade Federal de São Carlos
Via Washington Luiz, km 235 – São Carlos, SP – CEP: 13565-905
vpureza@dep.ufscar.br

RESUMO

Este trabalho apresenta um algoritmo GRASP com vistas à resolução de um problema de planejamento da produção encontrado na indústria de grãos eletrofundidos. O problema, previamente modelado como um programa linear inteiro misto combina decisões de seleção de processos e dimensionamento de lotes monoestágio. O método heurístico é proposto como alternativa à utilização do modelo, em particular, quando *softwares* de otimização não estão disponíveis ou quando o tamanho da instância compromete sua resolução por métodos exatos. Resultados obtidos com conjuntos de instâncias geradas a partir de um exemplo extraído de uma empresa do setor indicam que o algoritmo GRASP proposto é capaz de gerar soluções de boa qualidade.

PALAVARAS CHAVE. Programação da produção, Dimensionamento de lotes, Métodos heurísticos.

Área principal (PO na Indústria)

ABSTRACT

This paper presents a GRASP algorithm for solving a production planning problem faced by the electrofused grain industry. Such problem, previously modeled as a mixed integer linear program, combines decisions of process selection and mono-stage lot sizing. The heuristic method is proposed as an alternative to using the model, particularly when commercial optimization software is not available or when the size of the instance impedes applying exact methods. Results obtained with sets of instances randomly generated from an example extracted from a typical company of the sector indicate that the proposed GRASP algorithm is capable of generating good quality solutions.

KEYWORDS. Production planning. Lot sizing. Heuristics.

Main area (PO in Industries)

1. Introdução

O estudo neste trabalho é dirigido a um problema de planejamento e controle da produção encontrado na indústria de grãos eletrofundidos, a qual apresenta diversas unidades produtivas na região Sudeste do país e, em particular, no estado de São Paulo. Empresas deste setor compõem parte da indústria de transformação mineral que produz e comercializa vários tipos de produtos para aplicação nas indústrias de cerâmicos, abrasivos e refratários. O processo produtivo é caracterizado pelo consumo de grandes quantidades de energia elétrica, o que coloca o setor em uma situação difícil frente à crise energética dos últimos anos. Tal dificuldade, por outro lado, estimula a pesquisa de alternativas para otimização do processo e do planejamento da produção.

Neste contexto, consideramos o problema de programar a produção de uma carteira de pedidos por grãos eletrofundidos de maneira que a falta de produtos demandados e atrasos na entrega sejam minimizados. A cada dia do horizonte do planejamento, um dentre vários possíveis processos de produção deve ser selecionado, sendo que sua implementação resulta em um conjunto especificado de produtos em quantidades previamente determinadas. O problema pode ser visto, portanto, como de dimensionamento de lotes (*lot-sizing*) que ao invés de "lotes de produtos", utiliza "lotes de processos" para produzir um conjunto de produtos. Com vistas à resolução deste problema, propomos um algoritmo GRASP.

Segundo a teoria da complexidade computacional, a maioria dos problemas de dimensionamento de lotes são NP-Difíceis (HSU, 1983), o que torna sua resolução uma tarefa desafiadora. O desenvolvimento de ferramentas capazes de gerar programas de produção eficientes em tempo razoável é também estimulado pelo fato de programas de produção em situações reais precisarem ser modificados várias vezes devido a imprevistos ou pedidos urgentes (Luche *et al*.,2009).

Este artigo está organizado como se segue. Na Seção 2 é apresentado de forma breve o processo de produção no qual este trabalho se baseia. A discussão é baseada na planta de uma das empresas estudadas nesta pesquisa, mas também se aplica a outras empresas deste setor. Na Seção 3 é descrito o modelo algébrico do problema, e a Seção 4 apresenta o algoritmo GRASP. Na Seção 5, são analisados os resultados obtidos com o modelo e com o algoritmo proposto quando aplicados a uma instância real e a conjuntos de instâncias geradas aleatoriamente. Os experimentos também ilustram como a complexidade do problema cresce com o aumento do tamanho das instâncias. Finalmente, a Seção 6 discute as conclusões e próximos passos desta pesquisa.

2. Descrição do processo de produção

A planta considerada produz óxido de alumínio branco, e todos os produtos apresentam-se na forma de grãos cujos tamanhos variam desde centímetros até poucos micrômetros. A classificação é feita por um conjunto de peneiras vibratórias que separam os grãos por tamanho, e que pode ser montado com diferentes combinações de peneiras. A programação da classificação dos grãos deve ser feita em conjunto com a programação dos equipamentos de transformação (fornos, britadeiras e moendas). Uma dada combinação de peneiras e de regulagem de moendas e britadores define um processo de produção.

Cada processo produz um *mix* de produtos em quantidades especificadas. A Tabela 1 apresenta as quantidades (em quilos por dia) de 23 produtos (grãos eletrofundidos) que podem ser produzidos por 10 processos diferentes. Por exemplo, o processo 1 produz diariamente 2000 quilos do produto EC31_36, 600 quilos do produto EC31_120, 500 quilos do produto EC31_150, e assim por diante, totalizando 27700 quilos por dia (última linha da tabela). Note que diferentes processos podem ser produzir um mesmo produto em diferentes quantidades. Note também que um dado processo produz produtos que podem não ter sido demandados no período em questão. Como a seleção de um dado processo define os produtos e suas quantidades, é comum o carregamento de estoques de produtos ainda sem demanda por longos períodos de tempo até o

final do horizonte planejado. Limitações de capacidade das máquinas são consideradas no momento da elaboração dos processos de produção, ou seja, cada processo de produção leva em conta as restrições de capacidade dos equipamentos da linha de produção.

Tabela 1 – Quantidades (kg) de grãos eletrofundidos (linhas) produzidos pelos processos (colunas).

Produto	**Processo**									
	1	2	3	4	5	6	7	8	9	10
EC31_36	2000	2000	2000	2000	2000	2000	2000	2000	2000	
EC31_120	600		600		600		1000		600	
EC31_150	500			300	300			300	300	
EC31_180	300		300		300		300			
EC31_220	300			300						
EG52_120		500		500		500		500		10000
EG52_150		300	300			200	500		800	
EG52_180		300		300				300	1000	
EC31R3-8_1-4	2000	2000	2000	2000	1000	2000	2000	2000	3600	10000
EC31R3-5_7	5000	5000	5000	5000	4000	5000	5000	5000	4800	
EC31R5-16_4	2000	2000	2000	2000	1000	2000	2000	2000	5000	5000
EC31R_08_F	5000	5000	10000	8000	7000	7000	10000	10000		
EC31R_08_F1	6000				800		1500	1000		
EC31R_08_F2		400		500		1000		1000		
EC31R_08_F3					500		1000	1000		2000
EC31R_08_F4	2000		2000			1000				
EC31R_08_F5		900					1000			
EC31R_08_G			1000	1000	1000					
EC31R_08_G1		1000	1000		1000					
EC31R_08_G2						1000		1000	8000	
EC31R_08_G3	2000			5000		4000				
EC31R_08_G4		8000			7000					
EC31R_08_G5			1200	800	500		1000			
TOTAL	27700	27400	27400	27700	27000	25700	27500	26100	26100	27000

Em cada dia de trabalho, a empresa deseja realizar no máximo uma preparação de processo em função de altos tempos de *setup* das máquinas. Para cada dia, deve ser selecionado o processo que melhor atende a quantidade demandada de cada produto, sabendo-se que dificilmente todos os produtos demandados serão produzidos por este processo, e que outros produtos não demandados serão produzidos. A programação da produção de grãos está também condicionada às seguintes considerações:

- O horizonte de programação adotado em geral é de um mês, o que se resume normalmente a 19 períodos (dias) em que há produção na empresa;
- Novos pedidos de clientes podem ser aceitos ao longo de um horizonte de tempo já programado. Ou seja, dentro de uma programação definida, pode ser necessário refazer a programação para satisfazer a nova demanda.

3. Modelagem

O problema tratado neste trabalho pode ser visto como uma combinação do problema de seleção de processos e do problema de dimensionamento de lotes. Em problemas de seleção de processos, as demandas de produtos são conhecidas previamente ao longo de um horizonte de planejamento. A produção de cada produto pode resultar de vários "caminhos" alternativos (rotas, processos) pelos quais ele pode ser produzido. Os custos unitários e os recursos de produção

utilizados dependem do processo selecionado. Os recursos têm disponibilidade limitada em cada período (por exemplo, capacidade de produção em horas), de maneira que os produtos competem por estes recursos de acordo com o processo de produção escolhido. O problema consiste em determinar o quanto de cada produto deve produzido em cada processo a fim de minimizar os custos de produção, sujeito às restrições impostas por limitações de recursos e de demanda.

Assim como no problema de seleção de processos, em problemas de dimensionamento de lotes cada produto pode ser produzido por diferentes processos alternativos e os custos e recursos de produção dependem do processo escolhido. Entretanto, outras características também consideradas em problemas de dimensionamento de lotes incluem, por exemplo, a possibilidade de estocagem dos produtos a um dado custo em cada período. Além disso, o tempo e custo de preparação dos equipamentos envolvidos na produção de cada tipo de produto podem depender ou não da sequência programada (em seleção de processos não é realizado o sequenciamento dos processos escolhidos). O problema consiste em determinar o quanto produzir de cada produto em cada processo, de maneira a minimizar os custos de produção e estocagem, sujeito às restrições de limitação de recursos e atendimento da demanda.

Enquanto o problema de seleção de processos é de fácil resolução (classe P), o mesmo não pode ser dito sobre a maioria dos problemas de dimensionamento de lotes. Tal motivação aliada a sua relevância prática, justifica o fato da literatura especializada ser bastante vasta. Em particular, Drexl e Kimms (1997), Karimi *et al.* (2003) e Staggemeier e Clark (2001), apresentam revisões de variações do problema. Outros *surveys* sobre o problema são encontrados em Zhu e Wilhelm (2006), Gao *et al*. (2008), Quadt e Kuhn (2008) e Robinson *et al.*(2009). Estudos de problemas de dimensionamento e sequenciamento de lotes e métodos de solução são também encontrados em Bitran e Matsuo (1986), Karmarkar *et al.* (1987), Matta e Guignard (1989), Kimms (1996), Feng e Cheng (1998), Armentano *et al.* (1999), Clark e Clark (2000), Haase e Kimms (2000), Meyr (2000, 2002), Surie e Stadtler (2003), Fleszar e Hindi (2004), Toledo *et al.* (2007), Toledo *et al.* (2008), Poltroniere *et al.* (2008), Ferreira *et al.* (2009), Molina *et al.* (2009) e Tonaki e Toledo (2010).

Conforme discutido na Seção 1, o problema aqui tratado consiste em encontrar um programa de produção que minimize a falta, utilizando apenas um processo por período. Sua formulação matemática (Modelo MFP) é proposta em Luche *et al* (2009), e descrita a seguir. Deve-se ressaltar que o modelo admite como nulo o estoque inicial dos produtos.

Modelo MFP (Minimizar a Falta de Produção)

Variáveis:

x_{jt} 1 se o processo j é usado no período t, 0 caso contrário ($j = 1,.., J; t = 1,.., T$);

I_{it}^{-} falta do item i ao final do período t ($i = 1,.., m; t = 1,.., T$);

Parâmetros:

m quantidade de produtos;
J quantidade de processos de produção disponíveis;
T horizonte de programação (em períodos de produção);
a_{ij} quantidade do produto i produzido pelo processo j em um t $\forall$ ($i = 1,.., m; j = 1,.., J$);
d_{it} demanda do produto i no período t ($i = 1,.., m; t = 1,.., T$);

$$\text{Min } z = \sum_{t=1}^{T}\sum_{i=1}^{m} I_{it}^{-} \qquad (1)$$

s.a.

$$\sum_{t'=1}^{t}\sum_{j=1}^{J} a_{ij}x_{jt'} + I_{it}^{-} \geq \sum_{t'=1}^{t} d_{it'} \qquad \forall\ i, t \qquad (2)$$

$$\sum_{j=1}^{J} x_{jt} \leq 1 \qquad \forall\, t \qquad (3)$$

$$x_{jt} \in \{0,1\} \qquad \forall\ j, t \qquad (4)$$

$$I_{it}^{-} \geq 0 \qquad \forall\, i, t \qquad (5)$$

A função objetivo (1) minimiza a falta de produção dos produtos que possuem demanda. As restrições de demanda (2) incluem variáveis de falta para cada produto *i* em cada período *t*, de maneira que a quantidade total produzida de um produto até um dado período *t* mais a falta, deve ser maior ou igual à demanda acumulada daquele produto até o período *t*.

4. Um algoritmo GRASP para resolução do modelo MFP

Proposta por Feo e Resende (1989), GRASP é uma metaheurística probabilística cuja aplicação consiste na repetição de dois passos ou fases principais. Na primeira fase, uma solução é construída com base em uma função de avaliação gulosa, ou seja, escolhendo o próximo componente da solução com base no benefício local que este traz. A diferença primordial em relação a uma heurística de construção tradicional é que uma vez identificados os componentes candidatos mais atraentes com a função de avaliação gulosa, um deles é selecionado de forma aleatória para inclusão na solução parcial. A segunda fase destes algoritmos consiste na aplicação de melhorias a partir da solução inicial, geralmente por meio de buscas locais ou aplicação de outra metaheurística.

Esses dois passos constituem uma iteração GRASP. Quando a implementação é bem projetada, a execução de um grande número de iterações GRASP permite a geração de várias soluções distintas de alta qualidade.

4.1. Fase de construção

A construção de soluções é baseada na heurística construtiva também proposta em Luche *et al.* (2009). Naquela heurística, uma solução é construída sequencialmente, um período por vez. A escolha de um processo para um período *t* em particular é guiada por uma função que avalia quão bem cada processo atende as datas de entrega da demanda de todos os produtos do período *t* ao período *T*, assumindo que nenhuma produção é alocada do periodo *t*+1 até *T*. Define-se d'_{it} como a demanda devida do produto *i* no periodo *t*, isto é, a demanda do produto no período *t*, menos seu estoque no fim do período *t*-1 ($d'_{it} = d_{it} - I_{i,t-1}$) (onde $I_{i,t-1} < 0$ corresponde à falta acumulada de períodos anteriores). Assim, para cada processo *j* (*j* = 1,.., *n*), a função de avaliação F_{jt} é definida como:

$$F_{jt} = \sum_{i=1}^{m} \sum_{t''=t}^{T} Min\left\{ \frac{(a_{ij} - d'_{it''})}{(t''-t+1)^{v}}, 0 \right\} \qquad (6)$$

Note que F_{jt} compreende somente termos associados aos períodos com falta de produção ($a_{ij} - d'_{it} < 0$) e o impacto de cada termo na avaliação diminui conforme a distância de *t"* aumenta em relação ao período atual *t*. Este impacto é controlado pelo parâmetro *v*. Quanto maior é o valor de *v*, menor é a contribuição da falta em F_{jt}.

Os *p* processos melhor avaliados (conjunto *P*) em um dado período *t* são selecionados para análise. Cada processo em *P* é usado como um processo provisório (processo candidato) para o período *t*, e os estoques e as demandas devidas de todos os produtos de *t*=1 até *T* são atualizadas. A função de avaliação é então usada para determinar os processos nos períodos seguintes *t*+1 até *T*, um período por vez. Para cada período, o processo melhor avaliado é selecionado. Note que ao final deste processo, uma solução provisória completa de *t*=1 até *T* é gerada. Uma segunda solução provisória é obtida, desta vez, inicialmente atribuindo-se outro

processo em P a t=1 (note que quanto maior é o valor do parâmetro P, maior é o número de soluções provisórias e, consequentemente, maior é o esforço computacional requerido). A solução provisória que apresentar a menor falta de produção define o processo a ser utilizado em t. Os estoques e as demandas devidas do período t à T são atualizadas de acordo com esta decisão, e os processos atribuídos aos períodos t+1 a T na solução provisória correspondente são desconsiderados. Todo o procedimento é então repetido para o período t+1 a fim de determinar o processo a ele atribuído e, continuando de forma similar, para cada período restante do horizonte até que seja obtida uma solução completa.

Na fase construtiva do algoritmo GRASP aqui proposto, a função de avaliação (6) é também utilizada para medir o benefício da atribuição do processo j ao período t, e um processo j é atribuído a cada período t por vez. Entretanto, a seleção do processo j para o período t é feita de forma aleatória, considerando apenas um subconjunto de processos com melhor avaliação e armazenados em uma lista restrita de candidatos (LRC). O tamanho de LRC é controlado por um parâmetro p; note que se p=1, têm-se um comportamento puramente guloso e para p=J, um comportamento puramente aleatório. Os passos da fase construtiva são descritos na Figura 1.

1. Faça $d'_{it} = d_{it}$ para cada produto i em cada período t (i=1,..,m; t=1,..,T).
2. Para $t1$=1 até T:
2.1. Calcule $F_{j,t1}$ (equação (6.1)) para cada processo j (j = 1,..,J).
2.2. Faça P = lista de p processos com as melhores avaliações (LRC).
2.3. Atribua aleatoriamente um processo p de LRC ao período $t1$. Para cada produto i (i=1,..,m), atualize os estoques $I_{i,t1}$ no período $t1$ e as demandas devidas d'_{it} em todos os períodos restantes $t1$+1 à T.
3. Retorne a solução S resultante.

Figura 1 – Passos da fase de construtiva do algoritmo GRASP.

4.2. Fase de Melhoria

A fase de melhoria do algoritmo GRASP (Figura 2) consiste de procedimentos de busca local, aqui denominados k-opt, aplicados à solução gerada pela fase construtiva (solução S). A cada iteração da busca local gera-se uma nova solução por meio de operações de substituição dos processos correntemente designados aos períodos. A vizinhança N(S) de uma dada solução corrente S consiste das soluções obtidas com a substituição da sequência de processos designados em S do período t até o período $t + k - 1$ (onde $k \leq T - t + 1$) por uma sequência de processos diversa, onde k é um parâmetro de entrada. Todas as possíveis sequências de processos nestes k períodos são testadas, e aquela que melhor substitui os processos retirados (S') passa a ser a nova solução corrente S se apresentar um valor de falta menor que o de S. Caso contrário, S é um ótimo local e o procedimento é finalizado.

1. Seja S a solução obtida na fase construtiva. Faça nova_iteração = verdadeiro.
2. Enquanto nova_iteração = verdadeiro:
2.1. Para t = 1 até ($T - k + 1$):
2.1.1. Gere e avalie a vizinhança N(S) de S resultante da substituição da sequência de processos designados em S do período t até o período $t + k - 1$ por uma sequência de processos diversa.
2.1.2. Selecione a solução S' de N(S) tal que falta(S') ≤ falta(S''), ∀S'' ∈ N(S);
2.1.3. Se falta(S') < falta(S), faça $S = S'$ e nova_iteração = verdadeiro. Caso contrário, faça nova_iteração =falso.

Figura 2 – Passos da fase de melhoria do algoritmo GRASP.

Os passos do algoritmo GRASP são descritos na Figura 3. Note que opcionalmente, pode-se reaplicar movimentos *k*-opt ao ótimo local encontrado após a conclusão do laço do passo 2.1, desta vez, com valor de *k* maior que o anteriormente utilizado (passo 3).

1. Faça $S^* = \emptyset$, falta(S*) = ∞ e it (número de iterações) = 0.
2. Enquanto it $\leq$ itmax:
2.1. Aplique a fase construtiva (descrita na Figura 1), obtendo a solução S.
2.2. Aplique a fase de melhoria (descrita na Figura 2) a partir da solução do passo anterior, obtendo a solução (ótimo local) S. Se falta(S) < falta(S*), faça S* = S.
2.3. Faça it=it + 1.
3. (Passo opcional) Aplique a fase de melhoria a partir da solução S* com um valor de *k* maior que o valor utilizado no passo 2.2. Faça S* igual à solução resultante.
4. Retorne *S**.

Figura 3 – Passos gerais do algoritmo GRASP.

5. Resultados computacionais

Os experimentos foram realizados em um computador com processador Intel quadCore com 4 Gb de RAM e sistema operacional Windows XP. Como ferramenta computacional para resolução do modelo utilizou-se a linguagem de modelagem GAMS (BROOKE *et al*., 1992) com o *solver* CPLEX 11 (método *branch and cut*) (ILOG, 2009). O algoritmo GRASP foi implementado em linguagem *Object Pascal* (Borland Delphi 7) e executado na mesma máquina utilizada para resolver o modelo via GAMS/CPLEX.

5.1. Experimentos com dados da empresa

A instância real é caracterizada por 159 processos de produção disponíveis para a programação de 50 itens em 19 períodos de produção. Segundo informações da empresa, a programação realizada pelo seu PCP resultou em uma falta total de 13.450 kg. Apesar de não haver registros do tempo requerido para a obtenção desta solução, em geral os programadores da empresa necessitam de várias horas (até mesmo dias) para encontrar um programa de produção satisfatório para uma dada carteira de pedidos (Luche *et al*., 2009).

A solução obtida com a resolução exata do modelo MFP, por outro lado, apresentou falta total de 10.475 kg em um tempo computacional de 26 segundos. O algoritmo GRASP foi capaz de encontrar uma solução ótima alternativa em aproximadamente 21 minutos.

5.2. Experimentos com dados gerados aleatoriamente

Para melhor avaliar o desempenho do algoritmo GRASP e do GAMS/CPLEX, foram também realizados experimentos com conjuntos de instâncias geradas aleatoriamente a partir do exemplo acima da empresa. Estes experimentos foram realizados para verificar como o desempenho dos métodos é afetado ao se redistribuir a demanda do produto nos períodos, ao decrementar e incrementar a demanda em algumas porcentagens determinadas (variação de demanda), e ao incrementar os números de períodos de produção e demanda (variação de períodos). As médias dos resultados com variação na demanda e com variação no número de períodos são apresentadas, respectivamente, nas Tabelas 2 e 3. A segunda e a terceira coluna em cada tabela apresentam, respectivamente, o desvio relativo de otimalidade (*Gap*) e o tempo computacional médio utilizado pelo software de otimização GAMS/CPLEX para cada conjunto de instâncias. O tempo máximo de execução foi limitado em 3 horas (10.800 segundos). A quarta e a quinta coluna em cada tabela fornecem, respectivamente, a média dos desvios percentuais das soluções do algoritmo GRASP em relação aos resultados com o modelo e o tempo computacional médio (em segundos) utilizado.

5.2.1. Variação de Demanda

A Tabela 2 apresenta os resultados dos experimentos para 11 conjuntos de 10 exemplos (totalizando 110 exemplos) gerados aleatoriamente a partir do exemplo da empresa a fim de considerar mudanças nas demandas originais. No conjunto S, cada exemplo contém os mesmos m=50 produtos, n=159 processos e T=19 períodos (dias) do exemplo fornecido pela empresa. Estas características totalizam 3.021 variáveis inteiras no modelo MFP. Nos conjuntos D10-D50, as demandas dos produtos nas instâncias do conjunto S são reduzidas em 10, 20, 30, 40 e 50%, respectivamente. Da mesma forma, nos conjuntos I10-I50, as demandas dos produtos nas instâncias do conjunto S são incrementadas em 10, 20, 30, 40 e 50%.

Observa-se que o desvio percentual médio (DPM) das soluções do algoritmo GRASP em relação às soluções do *solver* CPLEX 11 varia entre 0,4% e 3,8%, com tempos de execução entre 1.193 e 1.815 segundos em cada conjunto de instâncias. A falta apresentada pelas soluções do algoritmo GRASP são, em média, apenas 1,5% superiores ao valor ótimo. Note, entretanto, que o tempo médio computacional é quase três vezes maior que o do GAMS/CPLEX.

Tabela 2 – Resultados com variação de demanda.

Conjunto	Modelo MFP		GRASP	
	Gap (%)	Tempo (s)	*DPM* (%)	Tempo (s)
D50	0	5	1,1	1.193
D40	0	8	0,4	1.206
D30	0	10	3,8	1.272
D20	0	27	0,5	1.374
D10	0	128	1,2	1.412
S	0	617	1,5	1.572
I10	0	1.134	1,0	1.695
I20	0	682	2,4	1.685
I30	0	593	1,7	1.724
I40	0	1.053	1,8	1.775
I50	0	1.443	1,3	1.815
Média	0	518	1,5	1.520

5.2.2. Variação de Número de Períodos

Na Tabela 3 são apresentados os resultados obtidos com 4 conjuntos de 10 exemplos (totalizando 40 exemplos) gerados aleatoriamente com base nas instâncias do conjunto S, discutido na seção anterior. Para cada instância de S, o número de períodos foi multiplicado por 2, 3, 4 e 5, resultando, respectivamente, nos conjuntos T2, T3, T4 e T5. A demanda do horizonte de planejamento original é então repetida exatamente em cada bloco de 19 períodos. Os conjuntos T2, T3, T4 e T5 totalizam no modelo MFP, um número de variáveis inteiras igual a 6.042, 9.063, 12.084 e 15.105, respectivamente.

Tabela 3 - Resultados com variação de períodos.

Conjunto	Modelo MFP		GRASP	
	Gap (%)	Tempo (s)	*DPM* (%)	Tempo (s)
T2	1,0	2.477	1,4	3.711
T3	1,6	3.378	1,4	6.588
T4	2,3	5.357	1,6	10.758
T5	2,9	7.372	1,6	10.800
Média	1,9	4.646	1,5	7.964

Os resultados mostram que se torna mais difícil resolver otimamente o modelo MFP conforme o número de períodos do horizonte de planejamento cresce. Isso é verificado pela redução substancial de certificados de otimalidade e o aumento do *gap*. Este resultado ilustra como a complexidade aumenta com o tamanho da instância.

Conforme observado na Tabela 3, o desvio percentual médio apresentado pelo GRASP em relação às soluções com o modelo é de 1,5%. Considerando que o *gap* médio do modelo é de apenas 1,9%, isso significa que assim como nas instâncias com variação da demanda, o GRASP alcançou resultados próximos do ótimo.

Em resumo, das duas estratégias discutidas nos experimentos, o modelo resolvido pelo *solver* CPLEX 11 obteve os melhores resultados em termos de qualidade de solução e tempo computacional.

6. Conclusões

Este trabalho abordou um problema de relevância prática no planejamento e controle da produção na indústria de grãos eletrofundidos. Um modelo de otimização é revisitado e um algoritmo GRASP é proposto para sua resolução. Os resultados obtidos permitiram inferir o desempenho do método heurístico relativo às soluções do *solver*, tanto em termos de qualidade de solução como de tempo computacional. Apesar dos resultados do algoritmo GRASP terem qualidade pouco inferior aos obtidos com o *solver*, o tempo computacional e a degradação relativamente pequena da qualidade das soluções em relação ao *solver* mostram que o algoritmo é competitivo e robusto, podendo ser empregado como alternativa aos pacotes de soluções comerciais como o GAMS/CPLEX. Assim, o algoritmo GRASP poderá ser considerado promissor se considerarmos as possibilidades de melhorias no método de construção ou de busca local.

Os resultados obtidos com uma instância real fornecida por uma das empresas estudadas indicaram que os métodos discutidos são capazes de produzir soluções melhores que as atualmente praticadas pela empresa. Devido aos tempos computacionais relativamente baixos para a aplicação em questão, tais abordagens permitem que sejam feitas várias simulações de programação da produção (explorando diferentes cenários), o que fornece flexibilidade e eficácia aos tomadores de decisão. Além disso, tais metodologias também facilitariam a aplicação de técnicas de horizonte rolante, e permitiriam aos departamentos de produção e vendas da empresa analisar rapidamente a incorporação de novos pedidos ao longo do horizonte de planejamento.

Dentre as perspectivas de pesquisa futura cabe destacar: (a) inclusão de decisões de mistura de dois ou mais produtos para produção de outro produto, (b) extensão dos modelos e algoritmos para consideração de incertezas nos parâmetros de entrada, de forma a serem tratados por técnicas de programação estocástica e otimização robusta, (c) desenvolvimento de um procedimento de geração de colunas para representar os diversos processos de produção de grãos eletrofundidos e sua incorporação em um método exato do tipo *branch-and-price* para obtenção

de soluções ótimas, e (d) desenvolvimento de implementações de meta-heurísticas alternativas, em particular, de otimização por colônia de formigas (ACO).

Agradecimentos

Esta pesquisa teve o apoio do CNPq (processos 200895/2005-2 e 522973/95-4).

Referências

ARMENTANO, V. A.; FRANÇA, P.M.; TOLEDO, F.M.B. (1999), A network flow model for the capacitated lot-sizing problem, *Omega International Journal of Management Science*, 27, 275-284.

BITRAN, G.R.; MATSUO, H. (1986), Approximation formulations for the single–product capacitated lot size problem, *Operations Research*, 34, 63-74.

BROOKE, A.; KENDRICK, D.; MEERAUS, A. GAMS: a user's guide (release 2.25)**.** The Scientific Press, San Francisco, 1992.

CLARK, A. R.; CLARK, S. J. (2000), Rolling-horizon lot-sizing when set-up times are sequence-dependent, *International Journal of Production Research*, 38, 2287-2307.

DREXL, A.; KIMMS, A. (1997), Lot sizing and scheduling: survey and extensions, *European Journal of Operational Research*, 99, 221-235.

FENG, H.; CHENG, H. (1998), Solving mixed integer programming production planning problems with setups by shadow price information, *Computers and Operations Research*, 25, 1027-1042.

FEO, T. A.; RESENDE, M.G. (1989), A probabilistic heuristic for a computationally difficult set covering problem, *Operations Research Letters*, 8, 67-71.

FERREIRA, D.; MORABITO, R.; RANGEL, S. (2009), "Solution approaches for the soft drink integrated production lot sizing and scheduling problem", *European Journal of Operational Research*, 196, 697-706.

FLESZAR, K.; HINDI, K.S. (2004), Solving the resource-constrained project problem by a variable neighbourhood scheduling search. *European Journal of Operational Research*, 155, 402-413.

GAO, L., ALTAY, N., ROBINSON, E.P. (2008), A comparative study of modeling and solution approaches for the coordinated lot-size problem with dynamic demand, *Mathematical and Computer Modelling*, 47, 1254-1263.

HAASE, K.; KIMMS, A. (2000), Lot sizing and scheduling with sequence-dependent setup costs and times and efficient rescheduling opportunities, *International Journal of Production Economics*, 66, 159-169.

HSU, W. (1983), On the general feasibility test of scheduling lot sizes for several products on one machine, *Management Science*, 29, 93-105.

ILOG. High – performance software for mathematical programming and optimization. http://www.ilog.com/products/cplex/, 5, 2009.

KARMARKAR, U. S.; KEKRE, S.; KEKRE, S. (1987), The dynamic lot sizing problem with startup and reservation costs, *Operations Research*, 35, 389-398.

KARIMI, B.; GHOMI, S.M.T.F; WILSON, J.M. (2003), The capacitated lot sizing problem: a review of models and algorithms, *Omega International Journal of Management Science*, 31, 365-378.

KIMMS, A. (1996), Multi-level, single-machine lot sizing and scheduling (with initial inventory), *European Journal of Operational Research*, 89, 86-99.

LUCHE, J.R.D.; MORABITO, R.; PUREZA, V. (2009), Combining process selection and lot sizing models for the production scheduling of electrofused grains. *Asia-Pacific Journal of Operations Research*, 26, 421-443.

MATTA, R.; GUIGNARD, M. Production scheduling with sequence independent changeover cost, *The Wharton School; University of Pennsylvania*, *Pennsylvania*, 1989.

MEYR, H. (2000), Simultaneous lot sizing and scheduling by combining local search whit dual reoptimization, *European Journal of Operational Research*, 139, 311-326.
MEYR, H. (2002), Simultaneous lot sizing and scheduling on parallel machines. *European Journal of Operational Research*, 139, 277-292.
MOLINA, F.; SANTOS, M. O.; TOLEDO, F. M. B.; Araujo, S. A. (2009), An approach using lagrangean/surrogate relaxation for lot-sizing with transportation costs. *Pesquisa Operacional*, 29, 269-288.
POLTRONIERE, S. C.; POLDI, K. C.; TOLEDO, F. M. B.; ARENALES, M. N. (2008), A coupling cutting stock-lot sizing problem in the paper industry, *Annals of Operations Research*, 157, 91-104.
QUADT, D., KUHN, H. (2008), Capacitated lot-sizing with extensions: a review, *4OR – A Quartely Journal of Operations Research*, 6, 61-83.
ROBINSON, P., NARAYANAN, A., SAHIN, F. (2009), Coordinated deterministic demand lot-sizing problem: A review of models and algoritms, *Omega*, 37, 3-15.
STAGGEMEIER, A. T.; A. R. CLARK. (2001), A survey of lot-sizing and scheduling models, *XXIII SBPO*, 938-947.
SURIE, C.; STADTLER, H. (2003), The capacitated lot-sizing problem with linked lot sizes, *Management Science*, 49, 1039-1054.
TOLEDO, C. F.; FRANÇA, P. M.; MORABITO, R.; KIMMS, A. (2007), Um Modelo de Otimização para o Problema Integrado de Dimensionamento de Lotes e Programação da Produção em Fábrica de Refrigerantes, *Pesquisa Operacional*, 27, 155-186.
TOLEDO, C. F.; FRANÇA, P. M.; MORABITO, R.; KIMMS, A. (2008), A multi-population genetic algorithm to solve the synchronized and integrated two-level lot-sizing and scheduling problem, *International Journal of Production Research*, 47, 3097-3119.
TONAKI, V. S.; TOLEDO, F. M. B. (2010), An approach for solving the lot-sizing problem of a market-driven foundry, *Journal of the Operational Research Society*, 61, 108-114.
ZHU, X.; WILHELM, W. E. (2006), Scheduling and lot sizing with sequence dependent setup: a literature review, *IIE Transactions*, 38, 987-1007.