Técnicas de *local branching* para o problema de abastecimento de linhas de montagem

Alysson A. A. Mendonça[1], **Mauricio C. de Souza**[1]

[1] Programa de Pós-Graduação em Engenharia de Produção
Universidade Federal de Minas Gerais (UFMG)

Resumo. *O problema de abastecimento de linhas de montagem pode ser observado na indústria automobilística, onde os sistemas de manufatura normalmente apresentam linhas de produção paralelas dedicadas à montagem de diferentes famílias de produtos. Propomos abordagens para a solução do problema baseadas na técnica de local branching, dentre as quais uma abordagem que explora um conjunto de soluções de elite gerados por GRASP e aproveita informações extraídas da relaxação linear de uma formulação de programação inteira. Resultados preliminares promissores mostram que a abordagem é competitiva e que tem potencial de refinamento em trabalhos futuros.*
PALAVRAS CHAVE*. abastecimento de linhas de montagem,* local branching*, otimização combinatória. Área de classificação principal PO na Indústria.*

Abstract. *The problem faced when feeding assembly lines can be observed in the automobile industry, in which manufacturing systems usually have parallel production lines dedicated to assembling different product families. We propose approaches for solving the problem based on the local branching technique, one of which explores a set of elite solutions generated by GRASP and also takes advantage of information extracted from solving the linear relaxation of an integer programming formulation. Promising preliminary results show a competitive approach which has refining potential on future works.*
KEYWORDS*. line feeding problem, local branching, combinatorial optimization. Main area OR in the industry.*

1 - Introdução

A padronização de métodos de trabalho em sistemas de manufatura é um importante fator a ser considerado em organizações que buscam atingir objetivos relacionados a alta produtividade ou redução de custo.

A organização do sistema de manufatura em linhas de produção e a utilização de metodologias como o *just-in-time* são exemplos de estratégias de padronização que podem ser verificados em diversos segmentos da indústria, como por exemplo a indústria automobilística.

Em sistemas de manufatura da indústria automobilística as linhas de produção geralmente são dispostas em paralelo, onde cada linha de produção é dedicada à montagem de uma família de produtos. Cada linha possui centros de trabalho dispostos em série. Um processo de abastecimento deve fornecer todos os itens necessários para os centros de trabalho das linhas de realizarem suas operações. Os itens são transportados da área de estoque para as linhas de produção em containeres que possuem tamanhos pré-definidos.

As políticas utilizadas no processo de abastecimento podem afetar significativamente os custos relacionados ao manuseio de itens entre a área de estoque e os centros de trabalho. Tipicamente, os custos relevantes neste contexto possuem componentes relacionados à estocagem e ao manuseio dos itens.

No presente estudo foram utilizadas informações reais de uma indústria automobilística localizada no Brasil. O processo de produção é realizado em cinco estágios: mecânica, prensagem, funilaria, pintura e montagem final. O estágio de montagem final é composto por quatro linhas de produção independentes que produzem veículos de acordo com as características de cada família de produtos. Em cada linha, veículos que estão em produção permanecem um período pré-definido em cada centro de trabalho, sendo este período denominado tempo de ciclo. O processo de abastecimento deve fornecer os itens necessários para as atividades de cada centro de trabalho considerando um horizonte de planejamento dividido em períodos. Por exemplo, um horizonte de planejamento considerando um intervalo de uma semana pode ser dividido em períodos de um dia. É possível formar estoque de itens na própria linha de produção, porém esta prática acarreta em custos que são calculados sobre a quantidade em estoque na linha de produção ao final de cada período.

O processo de abastecimento das linhas de produção da indústria possui as seguintes restrições operacionais:

1. Cada container transportado deve ser carregado com um único tipo de item.
2. Cada container transportado deve ser carregado com sua capacidade máxima, ou seja, a quantidade transportada por containeres de determinado tipo considerando um tipo de item específico é pré-definida.
3. Para cada tipo de item um único tipo de container deve ser utilizado para o transporte em todo o horizonte de programação.

Segundo Krajewski e Ritzman (1993) a restrição de utilização da capacidade máxima de todos os containeres é típica de sistemas *just-in-time*. Este conjunto de restrições é utilizado com o objetivo de diminuir variabilidades no processo de produção das linhas de produção.

O problema de abastecimento de linhas de montagem (LFP - *Line Feed Problem*) pode ser descrito como o problema de se obter a política de abastecimento de linhas de produção que minimiza os custos de transporte de itens e de estocagem na linha de produção, atendendo à demanda de todos os centros de trabalho a cada período e observando as restrições operacionais do processo de abastecimento.

Este estudo tem como objetivo avaliar a utilização de diferentes abordagens para solução do LFP baseadas na técnica de *local branching*. A seção 2 realiza uma breve revisão da literatura sobre o LFP e a técnica de *local branching*, a seção 3 detalha as abordagens propostas no trabalho, a seção 4 apresenta os resultados computacionais e a seção 5 conclui o trabalho apresentando também oportunidades para desenvolvimento de trabalhos futuros.

2 - Formulação matemática para o LFP

Segundo de Souza *et al.* (2008) o LFP é o problema encontrado ao decidir como empacotar itens nos containeres disponíveis com o objetivo de atender às necessidades dos centros de trabalho a um custo mínimo no horizonte de planejamento considerado. O LFP pode ser considerado um problema de empacotamento com tamanho de *bin* variável que possui características especiais: restrição da quantidade disponível de cada *bin* e uma estrutura de custos tal que o custo de cada *bin* varia de acordo com os itens empacotados. O mesmo trabalho mostra que a versão não-capacitada do LFP é NP-Difícil, tendo como um caso especial o *Change Making Problem* o qual é NP-Difícil mesmo em sua versão não-capacitada. Um caso especial do LFP, equivalente ao problema de decisão do *Maximum Cardinality Bin Packing Problem*, pode ser utilizado para demonstrar que encontrar uma solução viável para o LFP é um problema NP-Completo.

A formulação proposta para o LFP por de Souza *et al.* (2008) modela o problema com *I* tipos de item, *K* tipos de containeres e *T* períodos a serem considerados no horizonte de programação. São dados de entrada a demanda de cada item em cada período(d_{it}), o custo de transporte de cada tipo de container em cada período (b_{kt}), o custo de armazenamento na linha de produção por unidade de cada item (c_i) a capacidade de cada container considerando cada item (q_{ik}) e a quantidade máxima de cada tipo de container a ser utilizada simultaneamente em qualquer período (l_k). As variáveis de decisão atribuem quais tipos de containeres devem ser utilizados para transportar quais tipos de item (x_{ik}) e definem o número de viagens que o container do tipo k deve fazer para transportar o item i no período t (f_{ikt}). A constante M é um valor calculado para cada instância e corresponde ao maior número de viagens simultâneas necessárias para atender à demanda de algum dos itens no período de maior demanda utilizando o menor container disponível.

Com o objetivo de tornar a formulação matemática do LFP mais forte, da Cunha e de Souza (2008) apresentam uma desagregação da constante M para cada item, período e container. A nova família de constantes é dada por $M_{ikt} = \lceil \sum_{t'=t}^{T} d_{it'}/q_{ik} \rceil$. O trabalho apresenta também a desagregação das variáveis de atribuição para cada período e propõe uma família de desigualdades válidas para fortalecer a formulação do LFP.

No presente trabalho utilizamos uma formulação baseada nos trabalhos de Souza *et al.* (2008) e da Cunha e de Souza (2008) que obteve melhores resultados do que outras variantes baseadas nos mesmos trabalhos, como apresentado por Mendonça (2011). Na formulação as variáveis de estoque são eliminadas utilizando as substituições apresen-

tadas por Pochet e Wolsey (2006) e o cálculo do parâmetro M é refinado para cada item, container e período. A formulação resultante é apresentada a seguir:

$$Min \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{k=1}^{K} ((b_{kt} + (T - t + 1)c_i q_{ik}) f_{ikt}) \quad (1)$$

s.a.

$$\sum_{t'=1}^{t} \sum_{k=1}^{K} q_{ik} f_{ikt'} \geq \sum_{t'=1}^{t} d_{it'} \qquad \forall i, \forall t \quad (2)$$

$$\sum_{k=1}^{K} x_{ik} = 1 \qquad \forall i \quad (3)$$

$$\sum_{t=1}^{T} f_{ikt} - M_{ik1} x_{ik} \leq 0 \qquad \forall i, \forall k \quad (4)$$

$$f_{ikt} - M_{ikt} x_{ik} \leq 0 \qquad \forall i, \forall k, \forall t \quad (5)$$

$$\sum_{i=1}^{I} w_{ikt} x_{ik} \leq l_k \qquad \forall k, \forall t \quad (6)$$

$$f_{ikt} \in \mathbb{N}, x_{ik} \in \{0, 1\}, \forall i, \forall k, \forall t \quad (7)$$

O objetivo (1) minimiza o somatório dos custos de estocagem e dos custos de transporte. A família de restrições (2) garante que a quantidade em estoque somada à quantidade transportada é suficiente para atender a demanda em todos os períodos. A família de restrições (3) define que cada tipo de item será transportado por um único tipo de container em todos os períodos. A família de restrições (5) realiza o acoplamento entre as variáveis de atribuição e de transporte, sendo suficiente para a corretude do modelo, porém é utilizada também a família de restrições (4) com o objetivo de melhorar o *upper bound* obtido durante a computação. A família de restrições (6) limita o número de containeres disponíveis em qualquer período. A família de restrições (7) define o domínio das variáveis de decisão do modelo.

3 - *Local branching* aplicado ao LFP

O *local branching*, técnica proposta por Fischetti e Lodi (2003), é um método de solução exato de MIPs. O *local branching* controla algum outro algoritmo de solução de MIPs exato, priorizando a exploração parcial ou completa da vizinhança de soluções viáveis já identificadas.

Seja a formulação genérica de MIP proposta por Fischetti e Lodi (2003):

$$Min c^T x \quad (8)$$

s.a.

$$Ax \geq b \tag{9}$$
$$x_j \in \{0,1\} \forall j \in \beta \neq \emptyset \tag{10}$$
$$x_j \geq 0, inteiro \forall j \in \mathbb{G} \tag{11}$$
$$x_j \geq 0, \forall j \in \mathbb{C} \tag{12}$$

No modelo (8)-(12) o conjunto de variáveis $N = \{x_1, x_2, x_3, ..., x_n\}$ foi particionado nos subconjuntos β, $\mathbb{G}$ e $\mathbb{C}$ que correspondem respectivamente ao conjunto de variáveis binárias, conjunto de variáveis inteiras e conjunto de variáveis contínuas. O subconjunto β deve ser não-vazio, enquanto os subconjuntos $\mathbb{G}$ e $\mathbb{C}$ podem ser vazios.

Considerando uma solução viável de referência $\bar{x}$ para (8)-(12), o conjunto $\bar{S} = \{j \in \beta : \bar{x}_j = 1\}$ define o suporte binário de $\bar{x}$. Para qualquer parâmetro k inteiro positivo, a k-ésima vizinhança $N(\bar{x}, k)$ de $\bar{x}$ define um subproblema contendo o conjunto de soluções viáveis de (8)-(12) que satisfazem a seguinte restrição de *local branching*:

$$\delta(x, \bar{x}) : \sum_{j \in \bar{S}} (1 - x_j) + \sum_{j \in \beta \setminus \bar{S}} x_j \leq k \tag{13}$$

Na restrição (13) são contabilizadas as variáveis binárias que eram unitárias na solução de referência e se tornaram nulas, assim como as variáveis binárias que eram nulas e se tornaram unitárias. O número de trocas de valor nas variáveis binárias deve ser menor ou igual ao parâmetro inteiro k.

Para problemas onde a cardinalidade do suporte binário de qualquer solução viável de (8)-(12) é uma constante, a restrição (13) pode ser reescrita de forma simplificada:

$$\delta(x, \bar{x}) : \sum_{j \in \bar{S}} (1 - x_j) \leq k' \tag{14}$$

Considerando o LFP é possível utilizar a restrição (14) como restrição de *local branching*. Observando o modelo (1)-(7) é possível perceber que a cardinalidade do suporte binário das soluções viáveis para o problema é igual ao número de itens, pois de acordo com a família de restrições (3) cada item deve ser atribuído necessariamente a um único container durante todo o horizonte de planejamento.

O tamanho inicial das vizinhanças, dado pelo parâmetro k, deve ser suficientemente grande para possibilitar que a vizinhança da solução viável contenha soluções melhores, porém precisa ser suficientemente pequeno para que seja possível explorar a vizinhança adequadamente considerando um pequeno limite de tempo. Durante a computação o *local branching* pode alterar o valor do parâmetro k, aumentando o k para definir vizinhança maiores quando a vizinhança foi explorada completamente e se mostrou inviável ou diminuindo o k para definir vizinhança menores quando a exploração da vizinhança não encontrou solução viável após um limite de tempo definido previamente.

Neste trabalho foram realizadas duas implementações de *local branching* para o LFP, sendo a primeira denominada *Local Branching clássico* e a segunda *Local branching com intensificação baseada em soluções GRASP*

3.1 - Local Branching clássico

A implementação *local branching clássico* utilizou como base o framework *ALB++* proposto por Martinez e Cunha (2009). O *ALB++* é um framework desenvolvido em C++ com o objetivo de permitir a implementação de algoritmos baseados em *local branching* a problemas combinatórios, sendo baseado na técnica de *local branching* como proposta por Fischetti e Lodi (2003).

É utilizada como restrição de *local branching* para o LFP a equação (14), onde k é o tamanho da vizinhança, $\bar{S}$ é o suporte binário da melhor solução viável encontrada até o momento e j é a coordenada unidimensional de uma variável de atribuição dada pela mudança de coordenadas $j = i+(k*I)$. No contexto do LFP, a restrição de *local branching* indica que a exploração de cada vizinhança é limitada a até k mudanças de atribuição de item a container.

3.2 - Local branching com intensificação baseada em soluções GRASP

A implementação de *local branching com intensificação baseada em soluções GRASP* é baseada livremente nas idéias do *Path-Relinking*, técnica proposta inicialmente por Glover (1996) e discutida por Glover *et al.* (2000) e Resende e Ribeiro (2003), como uma estratégia de intensificação para explorar trajetórias definidas a partir de pares de soluções do conjunto elite.

A abordagem proposta realiza intensificação sobre pares de soluções do conjunto elite e diversificação observando informações extraídas da relaxação linear do problema. A abordagem contempla os seguintes passos:

1. Computação do conjunto de elite
2. Determinação do conjunto inicial de variáveis fixadas
3. Execução do local branching utilizando nova diversificação

No passo 1 é computado um conjunto de elite utilizando a heurística *GRASP* proposta para o LFP por de Souza *et al.* (2008). O *GRASP* foi executado para cada instância do LFP com um limite máximo de 500 iterações. Durante cada execução foi separado um conjunto de soluções denominado conjunto elite, contendo soluções a serem utilizadas como solução inicial do *local branching*. O conjunto elite é formado com no máximo 5 soluções. Uma nova solução é adicionada ao conjunto elite se a solução possui o melhor valor de função objetivo encontrado até a iteração atual do *GRASP* ou se as atribuições de item a container da solução são diferentes em ao menos 10% quando comparadas às atribuições de cada solução já presente no conjunto elite. Para o caso em que uma solução candidata é admitida no conjunto elite quando este já possui 5 soluções, é retirada do conjunto elite a solução de pior função objetivo, mantendo a cardinalidade do conjunto elite constante.

Utilizando os critérios apresentados anteriormente, é possível que para determinada instância a execução do *GRASP* finalize com um conjunto elite contendo uma única solução. Este comportamento indica que a solução encontrada na primeira iteração do *GRASP* possui o melhor valor de função objetivo dentre todas as soluções encontradas durante a execução. Adicionalmente, o comportamento indica que a partir da segunda iteração não foram encontradas soluções com diferença de ao menos 10% nas atribuições de item a container quando comparadas à solução encontrada na primeira iteração.

Como última etapa da formação do conjunto elite, para cada solução a abordagem aproveita as atribuições de item a container (família de variáveis x_{ik}) fixando seus valores e chama o CPLEX para encontrar os valores ótimos das variáveis de transporte (família de variáveis f_{ikt}), sendo esta computação realizada tipicamente em menos de um segundo para cada solução. A solução com valores ótimos das variáveis de movimentação então entra no conjunto elite substituindo a solução do conjunto na qual foi baseada.

No passo 2 são utilizadas idéias do *Path-Relinking* para determinar uma lista de variáveis a serem fixadas inicialmente. O procedimento cria uma lista de variáveis de atribuição a serem fixadas de forma a intensificar a exploração do espaço de busca em trajetórias definidas a partir de pares de soluções do conjunto elite. Para cada par de soluções do conjunto elite, é adicionada na lista de variáveis a serem fixadas as variáveis que possuem o mesmo valor atribuído nas duas soluções. A fixação de variáveis é realizada adicionando restrições ao modelo da forma $x_{ik} = \hat{x}$, onde $\hat{x}$ é o valor atribuído à variável x_{ik} nas soluções.

No passo 3 são utilizadas idéias do *Relaxation induced neighbourhood search*, técnica de fixação de variáveis proposta por Danna *et al.* (2005), para criar uma lista de itens que devem ter suas variáveis de atribuição liberadas. A lista de itens é ordenada por uma função que tenta quantificar a expectativa de melhoria da função objetivo se as variáveis de atribuição associadas ao item forem deixadas livres para a determinação de valor pelo *solver*.

O procedimento de criação da lista contendo itens que devem ter suas variáveis de atribuição liberadas considera que variáveis binárias que tiveram um valor atribuído na solução da relaxação linear mais próximo de $0,5$ possuem uma expectativa maior de melhoria da solução objetivo quando forem deixadas livres, dado que variáveis que possuem valor próximo de 0 ou 1 na relaxação linear e na solução viável provavelmente já levam a uma boa solução. O procedimento cria uma lista de itens candidatos à relaxação contendo os itens em ordem não-decrescente de acordo com a seguinte função $deltaRL_i$:

$$deltaRL_i = Min(|\bar{x}_{ik} - 0,5|) \tag{15}$$

Na função (15) i é o número do item, k é o número do container e $\bar{x}_{ik}$ é o valor atribuído pela relaxação linear à variável de atribuição do item i ao container k.

Como descrito anteriormente, quando uma vizinhança é explorada completamente e se mostra inviável o parâmetro k é aumentado para permitir a exploração de uma vizinhança maior. Quando a exploração da nova vizinhança não identifica soluções viáveis, parte das variáveis fixadas no passo 2 é liberada através da remoção das restrições que fixavam estas variáveis. Cada vez que esta diversificação ocorre o procedimento libera as restrições associadas a 10% dos itens, utilizando a ordem de liberação de itens definida através lista ordenada pela função (15).

A última etapa da abordagem é a execução do *local branching* utilizando a diversificação detalhada no passo 3.

4 - Resultados computacionais

Os experimentos desta seção foram realizados em uma estação com processador Intel Core2Duo 1.3Ghz, 4gb de memória Ram, sistema operacional Ubuntu Linux 10 versão

64 bits. Todas as implementações foram realizadas em linguagem C++ utilizando o compilador G++ versão 4.3. O pacote de otimização CPLEX utilizado para resolver os subproblemas nas implementações e para comparação foi o CPLEX versão 12.2. As implementações foram avaliadas utilizando as instâncias apresentadas por da Cunha e de Souza (2008), considerando custos de transporte variáveis.

4.1 - Resultados do local branching clássico

Os testes da implementação *local branching clássico* foram baseados em uma parametrização obtida empiricamente, utilizando uma vizinhança de tamanho (K) igual a 5, realização de no máximo 20 diversificações, limite de tempo 300 segundos para exploração de cada vizinhança e limite de tempo de execução para cada instância de 7200 segundos. Os resultados do CPLEX foram obtidos utilizando a melhor parametrização encontrada empiricamente, utilizando as opções *mipemphasis=4* e *nodesel=0*.

Instância	Local branching clássico	CPLEX
tb191b_111	458246	450328
tb191b_113	429765	427576
tb191b_211	520211	503689
tb191b_213	453683	452881
t191_113	488095	484971
t191_121	465551	465227
t191_213	529025	518152
t191_222	518864	503489

Tabela 1: Resultados da implementação *local branching clássico*

A Tabela 1 apresenta a instância avaliada, o melhor *upper bound* obtido durante a execução do *local branching* clássico e o melhor *upper bound* obtido durante a execução do CPLEX.

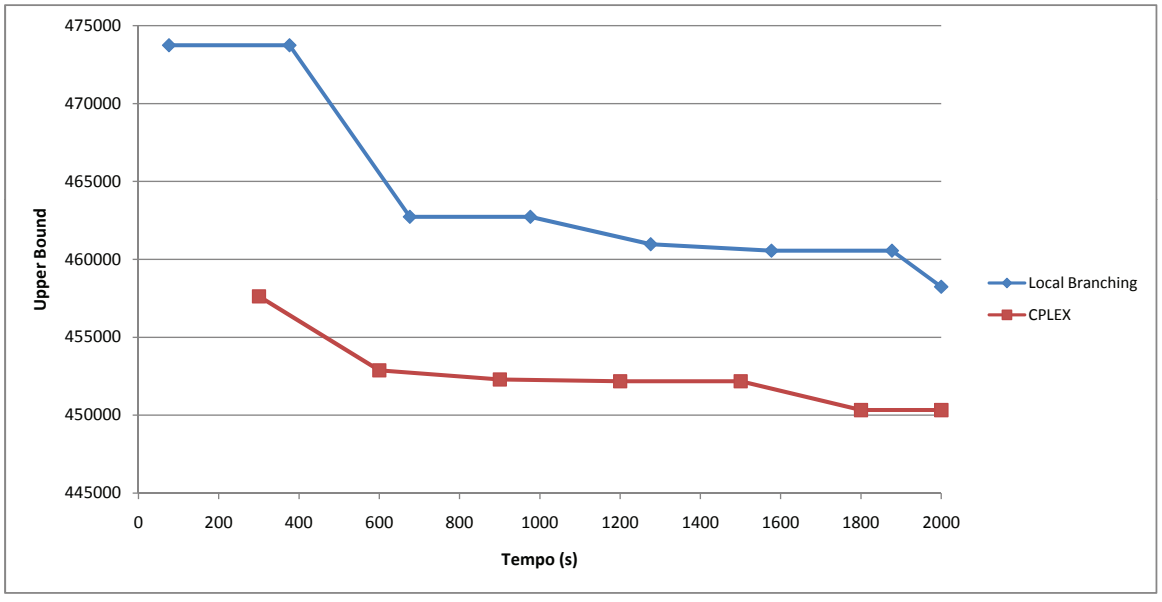

Figura 1: Convergência do *local branching* clássico na instância tb191b_111

Para todas as instâncias avaliadas o *upper bound* obtido com a utilização do *local branching* clássico foi pior do que o *upper bound* com a utilização do CPLEX utilizando o mesmo limite de tempo. A Figura 1 apresenta o gráfico de convergência do *upper bound* para uma das instâncias avaliadas. O comportamento se repetiu para os testes efetuados com outras instâncias.

Uma hipótese para explicar a convergência de *upper bound* observada na implementação *local branching* clássico seria de que o grande número de variáveis binárias no problema dificulta a efetiva exploração de vizinhanças pelo método. Quando parametrizado para explorar vizinhanças pequenas (parâmetro k), pouca melhoria do *upper bound* era observada após a exploração de cada vizinhança, enquanto que a parametrização de exploração de vizinhanças grandes leva o algoritmo a não determinar a otimalidade de cada vizinhança no tempo limite de exploração.

4.2 - Resultados do local branching com intensificação baseada em soluções GRASP

A implementação de *local branching com intensificação baseada em soluções GRASP* foi testada utilizando como parâmetros uma vizinhança de tamanho (k) igual a 5, tempo limite para exploração de vizinhanças de 60 segundos e limite de tempo de execução para cada iteração de 400 segundos, sem limite máximo de realização de diversificações.

Foram testadas duas estratégias para fixar variáveis. Na primeira, são comparadas para fixação as variáveis de atribuição das duas soluções do conjunto de elite como o melhor valor de função objetivo. Já na segunda, são comparadas para fixação as variáveis de atribuição das duas soluções mais distantes, sendo a distância entre duas soluções de elite dada pelo número de variáveis de atribuição de item a container que são diferentes nas duas soluções. Nas duas estratégias a solução do par considerado que possui melhor *upper bound* é utilizada como solução inicial para a computação.

A Tabela 2 está dividida em seções. A seção *Entrada* apresenta a identificação de cada instância (Instância), número de soluções no conjunto elite (#Elite) e o *upper bound* da melhor solução do conjunto elite (UB Elite). A seção *LB (melhor par)* contém os resultados obtidos utilizando a estratégia que considerou as duas soluções de melhor função objetivo no conjunto elite, sendo apresentados o *upper bound* (UB), número de variáveis fixadas inicialmente (#Fix) e tempo decorrido quando o *upper bound* foi identificado (t). A seção *LB (par mais distante)* contém os resultados obtidos utilizando a estratégia que considerou as duas soluções mais distantes entre si no conjunto elite, sendo apresentados o *upper bound* (UB), número de variáveis fixadas inicialmente (#Fix) e tempo decorrido quando o *upper bound* foi identificado (t).

A seção *CPLEX 400s* contém os resultados obtidos pelo CPLEX após 400 segundos de computação, sendo apresentados o *upper bound* (UB), o *lower bound* (LB) e o GAP relativo percentual (Gap).

As colunas *#Fix* da Tabela 2 apresentam o número de variáveis da família x_{ik} fixadas inicialmente antes do início da execução do *local branching*.

Em onze instâncias o conjunto elite continha uma única solução. Este comportamento indica que a solução encontrada na primeira iteração do *GRASP* possui o melhor valor de função objetivo dentre todas as soluções encontradas durante a execução. Adicionalmente, o comportamento indica que a partir da segunda iteração não foram encontradas soluções com diferença de ao menos 10% nas atribuições de item a container quando

Entrada			LB (melhor par)			LB (par distante)			CPLEX 400s		
Instância	#Elite	UB Elite	UB	#Fix	t (s)	UB	#Fix	t (s)	UB	LB	Gap (%)
tb191b_111	1	471904	471283	573	269	471283	573	269	457069	407703	10,8
tb191b_112	1	429062	427580	573	349	427580	573	349	433284	395236	8,8
tb191b_113	1	429062	427646	573	392	427646	573	392	428652	390683	8,9
tb191b_121	4	475159	475159	569	0	475095	565	380	471195	461586	2,0
tb191b_122	5	513401	513182	563	300	513323	555	9	512070	506269	1,1
tb191b_123	5	525054	523746	551	392	524397	545	11	526799	507581	3,6
tb191b_211	4	518030	514226	569	322	515198	565	301	507444	463994	8,6
tb191b_212	1	455319	454570	573	356	454570	573	356	454442	419886	7,6
tb191b_213	1	454341	453926	573	387	453926	573	387	453449	420441	7,3
tb191b_221	5	514077	514036	559	1	514036	555	1	513391	498886	2,8
tb191b_222	5	552749	552749	573	3	552392	561	100	552201	540521	2,1
tb191b_223	5	563463	563187	555	398	562702	541	400	564364	544237	3,6
t191_111	1	433942	433488	573	384	433488	573	384	437501	391522	10,5
t191_112	4	452228	449351	569	362	449250	563	87	446169	409288	8,3
t191_113	5	511517	509367	573	400	505124	563	391	495149	444568	10,2
t191_121	1	465279	465279	573	-	465279	573	-	465252	460761	1,0
t191_122	1	474479	474479	573	-	474479	573	-	466039	460738	1,1
t191_123	5	479061	478915	565	3	477283	561	380	475252	463875	2,4
t191_211	1	469147	468212	573	396	468212	573	396	457564	419012	8,4
t191_212	3	479363	477346	567	400	477630	565	263	476662	440297	7,6
t191_213	3	540590	534720	559	287	537562	557	400	529707	478614	9,6
t191_221	1	503053	502942	573	385	502942	573	385	503271	496411	1,4
t191_222	1	552734	552424	573	3	552424	573	3	505361	496370	1,8
t191_223	5	516833	514787	569	400	515142	559	207	510878	499742	2,2

Tabela 2: Resultados da implementação *local branching com intensificação baseada em soluções GRASP*

comparadas à solução encontrada na primeira iteração. Para estes casos os resultados das seções *LB (melhor par)* e *LB (par mais distante)* na Tabela 2 são os mesmos, correspondendo aos valores obtidos em uma única execução onde inicialmente todas as variáveis da solução do conjunto elite são fixadas.

Em treze instâncias o conjunto de elite continha mais do que duas soluções. Para estas instâncias foram realizados testes utilizando as duas estratégias de fixação de variáveis. Considerando este subconjunto das instâncias, em uma instância (tb191b_221) as duas estratégias obtiveram o mesmo *upper bound*, em seis instâncias (tb191b_122, tb191b_123, tb191b_211, t191_212, t191_213, t191_223) a estratégia de fixação de variáveis que considera o melhor par de soluções do conjunto elite obteve o melhor *upper bound* do *local branching*, enquanto nas demais seis instâncias (tb191b_121, tb191b_222, tb191b_223, t191_112, t191_113, t191_123) a estratégia de fixação de variáveis de melhor *upper bound* obteve os melhores limites.

Em seis instâncias o *upper bound* obtido utilizando uma das estratégias de fixação de variáveis foi melhor do que o obtido pelo CPLEX após 400 segundos de computação. Considerando o subconjunto destas seis instâncias, não foi possível identificar um padrão de parâmetros ou condições que pudessem ser identificados como determinantes para uma convergência mais rápida do *local branching*.

5 - Conclusão

O problema de abastecimento de linhas de montagem possui aplicação direta na indústria automobilística, possuindo importantes implicações econômicas e operacionais no contexto do processo de produção. O problema possui aspectos que se assemelham a vários problemas clássicos como o problema de dimensionamento de lotes não capacitado e o problema de empacotamento com tamanho de *bin* variável, apresentando assim uma mescla de elementos relativos ao sequenciamento e ao dimensionamento de lotes.

O trabalho apresentou abordagens para a aplicação da técnica de *local branching* ao problema de abastecimento de linhas de montagem.

A abordagem *local branching clássico*, mais próxima do método de *local branching* originalmente apresentado na literatura, obteve resultados desfavoráveis quando comparada à utilização do pacote de solução de *MIPs* comercial pois foi observada convergência mais lenta do que o pacote em todas as instâncias testadas.

A abordagem *local branching com intensificação baseada em soluções GRASP*, proposta no trabalho, utiliza um conjunto elite formado à partir de soluções obtidas com a utilização de um algoritmo GRASP. A abordagem obteve resultados favoráveis em parte das instâncias quando comparado à utilização do pacote de solução de *MIPs* comercial.

São destacadas como oportunidades promissoras para o desenvolvimento de trabalhos futuros o refinamentos da estratégia utilizada na criação do conjunto elite, elaboração de novas estratégias para a fixação e liberação de variáveis e a utilização de abordagens diferentes para a solução do problema de abastecimento de linhas de montagem, como por exemplo, a utilização de técnicas baseadas em programação dinâmica.

Referências

da Cunha, A.S. & de Souza, M.C. (2008). Stronger upper and lower bounds for a hard batching problem to feed assembly lines. *Electronic Notes in Discrete Mathematics*, **30**, 159–164.

Danna, E., Rothberg, E. & Pape, C.L. (2005). Exploring relaxation induced neighborhoods to improve mip solutions. *Mathematical Programming*, **102**, 71–90.

de Souza, M.C., de Carvalho, C.R.V. & Brizon, W.B. (2008). Packing items to feed assembly lines. *European Journal of Operational Research*, **184**, 480–489.

Fischetti, M. & Lodi, A. (2003). Local branching. *Mathematical Programming*, **98**, 23–47.

Glover, F. (1996). Tabu search and adaptive memory programing - advances, applications and challenges. In *Interfaces in Computer Science and Operations Research*, 1–75, Kluwer.

Glover, F., Laguna, M., Martí, R., Womer, D.K., A, F.G., B, M.L. & C, R.M. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, **39**, 653–684.

Krajewski, L.J. & Ritzman, L.P. (1993). *Operations Management: Strategy and Analysis*. Addison-Wesley, Reading, 3rd edn.

Martinez, L.C. & Cunha, A.S. (2009). Um arcabouço local branching para problemas de otimização combinatória aplicado ao problema da Árvore de custo mínimo com k arestas. In *XLI Simposio Brasileiro de Pesquisa Operacional*, SBPO, Porto Seguro, Brasil.

Mendonça, A.A.A. (2011). *Modelos e técnicas de local branching para o problema de abastecimento de linhas de montagem*. Dissertação de mestrado, Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Minas Gerais, Belo Horizonte, Brasil.

Pochet, Y. & Wolsey, L.A. (2006). *Production Planning by Mixed Integer Programming*. Springer-Verlag, New York, 1st edn.

Resende, M.G.C. & Ribeiro, C.C. (2003). GRASP and path-relinking: Recent advances and applications. In *Proceedings of the Fifth Metaheuristics International Conference (MIC2003)*, T6–1 – T6–6, Toshihide Ibaraki and Yasunari Yoshitomi, Kyoto, Japão.