An Application of the Local Branching to the Identical Parallel Machines Scheduling Problem

T. Schimidt, C. Scarpin, G. Loch, and N. da Silva

***Abstract*—In this paper, we consider the Identical Parallel Machine Scheduling Problem, with sequence-dependent setup time and the aim of minimizing makespan. We present two solution approaches: mathematical formulation of Mixed Integer Linear Programming (MILP) and the exact Local Branching method. The aim of this paper is the application and comparison of performance between the two approaches and the analysis of the convergence of each method. This analysis was based on the variation of setup times (between 10% and 50% of the processing time of the task in a machine, depending on the test performed), number of tasks (20, 30, 40, 50 and 100) and the number of machines available (3 and 5 machines). Both approaches were tested with a computational time of one hour in each test. As a consequence, we compared the optimality GAP for all instances by calculating the Lower Bound of each generated problem. The results obtained for instances with 30 tasks or more indicate that the Local Branching method presented better results when having a more significant number of machines. This gain increases as there is an increase in the number of tasks and machines. For smaller problems and with fewer parallel machines, the MILP approach presents lower GAP values.**

***Index Terms*—Lot sizing and scheduling, Buffer between stages, Time lags, Decomposition heuristics.**

I. INTRODUÇÃO

O sequenciamento de produção é uma ferramenta amplamente utilizada para a tomada de decisões em diversos problemas reais [1] [2], que ocorrem em sistemas de produção e de serviços [3]. Os gestores responsáveis pelo planejamento deste sistema não almejam apenas minimizar os custos do processo, como também melhorar a oferta de atendimento, sem perder os prazos estabelecidos [4].

Dentre os ambientes produtivos comumente encontrados na literatura, este estudo aborda um problema de sequenciamento em máquinas paralelas idênticas (SMPI), no qual o tempo de preparação da máquina (tempo de *setup*) depende da sequência estabelecida [5].

This work was supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior under grants 1708592 and 1615501.

T. M. P. Schimidt, Grupo de Tecnologia Aplicado à Otimização - Universidade Federal do Paraná, Curitiba, Paraná, Brasil (e-mail: talitapschimidt@gmail.com).

C. T. Scarpin, Grupo de Tecnologia Aplicado à Otimização - Universidade Federal do Paraná, Curitiba, Paraná, Brasil (e-mail: cassiusts@gmail.com).

G. V. Loch, Grupo de Tecnologia Aplicado à Otimização - Universidade Federal do Paraná, Curitiba, Paraná, Brasil (e-mail: gustavo.gvalentim@gmail.com).

N. C. O. da Silva, Instituto Federal do Rio Grande do Sul, Erechim, Rio Grande do Sul, Brasil e Grupo de Tecnologia Aplicado à Otimização - Universidade Federal do Paraná, Curitiba, Paraná, Brasil (e-mail: ncosilva2@gmail.com).

O sequenciamento em máquinas paralelas idênticas pode ser considerado um processo de dois passos: primeiramente, busca-se estabelecer quais tarefas devem ser alocadas em alguma máquina disponível e, em seguida, determina-se a sequência dos trabalhos designados a cada máquina [6]. A sequência apresentada pode interferir no tempo de *setup*, já que relaciona o trabalho a ser processado àquele que está em processo de finalização na sequência [7].

A literatura traz diversas pesquisas tratando deste tema, em especial, para o caso em que o objetivo é minimizar o tempo total de conclusão das tarefas ou *makespan* [8]. Segundo a notação clássica de [9], o problema em questão pode ser denotado por, $P_m \ |s_{ij}| \ C_{max}$, onde m determina o número de máquinas paralelas, s_{ij} é o tempo de *setup* entre as tarefas i e j (quando i é imediatamente predecessora a j) e C_{max} é o tempo de conclusão de todas as tarefas (*makespan*).

O tempo de *setup* representa o tempo necessário para a preparação de um recurso que realizará alguma operação. Desse modo, a redução do tempo de *setup* pode auxiliar na contenção de despesas, aumento de produção, menor estoque, entregas mais rápidas, redução dos prazos de entrega, maior competitividade e maior satisfação dos clientes [10]. O estudo de [11] propõe uma abordagem heurística baseada na geração aleatória de várias sequências e seleciona o melhor sequenciamento obtido. Neste problema, os sequenciamentos em uma única máquina e em máquinas paralelas idênticas estão sujeitos ao tempo de *setup* dependente da sequência e a data de lançamento das tarefas, com o objetivo de minimizar o *makespan*. No trabalho de [7], os autores aperfeiçoaram o método heurístico de [12] por meio de técnicas de Programação Linear para o problema de sequenciamento em máquinas paralelas idênticas, com tempo de *setup* dependente da sequência e divisão dos trabalhos (um item pode ser dividido em várias seções e cada seção pode ser processada por qualquer máquina).

A pesquisa de [13] apresenta um modelo de Programação Linear Inteira Mista para minimizar o *makespan* e o atraso total para o problema de sequenciamento em máquinas paralelas idênticas com tempo de *setup* dependente da sequência. Uma nova versão do Algoritmo Genético de Classificação não Dominada (NSGA-II) e uma abordagem exata baseada no método de duas fases foram desenvolvidas para a resolução desse problema. O Algoritmo Genético, também utilizado para resolver um problema de Máquinas Paralelas não relacionadas, mostra resultados satisfatórios com o objetivo de minimizar o *makespan* no trabalho de [14]. Em [15], os autores compararam os resultados obtidos pelas metaheurísticas Busca Tabu e Algoritmo Memético. O objetivo, neste caso, é minimizar o tempo total de conclusão

das tarefas para máquinas paralelas idênticas. Por sua vez, [16] apresentaram um modelo de Programação Linear Inteira Mista e desenvolveram duas metaheurísticas para o problema de sequenciamento com um servidor: o *Simulated Annealing* (SA) e o Algoritmo Genético (AG). Neste trabalho, os autores compararam os métodos propostos com as regras de prioridade *Shortest Processing Time* (SPT) e *Longest Processing Time* (LPT) e concluíram que o AG concedeu os melhores resultados obtidos.

Diversas aplicações podem ser encontradas nos ambientes de produção, serviço e informação [10]. Operações em uma unidade de aviação militar com o objetivo de realizar missões de voo [17]; a fabricação de produtos de couro do cloreto de polivinil [18]; e o processo de corte e costura para a confecção de móveis estofados [19] são exemplos de aplicações envolvendo o sequenciamento em máquinas paralelas idênticas.

O problema de sequenciamento em máquinas paralelas idênticas é considerado *NP-hard* [20]. O que torna a obtenção de sua solução mais complexa conforme o aumento do número de tarefas realizadas e máquinas disponíveis. Deste modo, busca-se um procedimento que estabeleça soluções ótimas ou próximas da ótima em um tempo computacional aceitável.

Diante deste contexto, este trabalho tem como objetivo a aplicação e avaliação do método exato *Local Branching* na solução de um problema de SMPI, que busca explorar partes do espaço de solução estrategicamente definidas e controladas por uma estrutura de ramificação. Essas ramificações são responsáveis pela geração de uma nova restrição no problema, limitando o espaço de busca.

Cabe observar que não se encontra na literatura a aplicação deste método para a solução do problema tratado neste trabalho, no entanto, para outras classes de problemas, de modo geral, proporciona uma boa convergência para as soluções encontradas, mas não garante a otimalidade na maioria dos casos [21], [22]. Deste modo, a qualidade das soluções obtidas pela metodologia será avaliada de acordo com um *Lower Bound* estabelecido para o problema de SMPI.

A sequência deste artigo está organizada da seguinte forma: na próxima seção, descreve-se detalhadamente as características do problema abordado, apresenta-se a formulação do modelo matemático e o *Lower Bound* referente ao SMPI. O método *Local Branching*, bem como a heurística responsável pela inicialização do processo de ramificação, é apresentado na seção III. A abordagem proposta, a qual é avaliada por meio de testes computacionais realizados em instâncias geradas aleatoriamente, e os resultados obtidos são descritos na seção IV. Por fim, as considerações finais e sugestões para trabalhos futuros são apresentadas na seção V.

II. MODELO DE PROGRAMAÇÃO LINEAR INTEIRA MISTA

O modelo de Programação Linear Inteira Mista (PLIM) aplicado nesta pesquisa é adaptado de [5], para o problema de sequenciamento em máquinas paralelas idênticas com tempo de *setup* dependente da sequência. O objetivo do problema é minimizar o *makespan*.

As premissas utilizadas no modelo são: cada tarefa pertencente ao conjunto $N = \{1, \dots, n\}$ deve ser realizada por uma única máquina pertencente ao conjunto $M = \{1, \dots, m\}$. Nenhuma tarefa pode ser interrompida durante a sua execução, ou seja, o processo de preempção não é permitido durante o sequenciamento; o tempo de processamento (p_j) da tarefa j e o tempo de *setup* (s_{ij}) entre as tarefas i e j são os parâmetros inteiros e não-negativos definidos a priori; as variáveis de decisão são C_{ik} e C_{max}, que representam, respectivamente, o instante de término da tarefa i na máquina k e o tempo total de conclusão das tarefas (*makespan*); e a variável binária x_{ijk} recebe o valor 1 caso a tarefa i preceda imediatamente a tarefa j e ambas sejam realizadas na máquina k. Caso contrário, a variável x_{ijk}recebe o valor 0.

A função objetivo e as restrições do problema de SMPI são apresentadas, como segue:

$$min\ C_{max} \tag{1}$$

Sujeito a:

$$\sum_{k \in M} \sum_{\substack{i \in \{0\} \cup \{N\} \\ i \neq j}} x_{ijk} = 1 \qquad \forall j \in N \tag{2}$$

$$\sum_{k \in M} \sum_{\substack{j \in N \\ i \neq j}} x_{ijk} \leq 1 \qquad \forall i \in N \tag{3}$$

$$\sum_{j \in N} x_{0jk} \leq 1 \qquad \forall\ k \in M \tag{4}$$

$$\sum_{\substack{h \in \{0\} \cup \{N\} \\ h \neq i, h \neq j}} x_{hik} \geq x_{ijk} \qquad \forall i, j \in N, i \neq j, \ \forall k \in M \tag{5}$$

$$C_{jk} + M\left(1 - x_{ijk}\right) \geq C_{ik} + s_{ij} + p_j \qquad \forall i \in \{0\} \cup \{N\}, \ \forall j \in N, i \neq j, \ \forall k \in M \tag{6}$$

$$C_{0k} = 0 \qquad \forall k \in M \tag{7}$$

$$C_{max} \geq C_{ik} \qquad \forall i \in N, \ \ \forall k \in M \tag{8}$$

$$C_{ik} \geq 0 \qquad \forall i \in N, \ \ \forall k \in M \tag{9}$$

$$x_{ijk} \in \{0,1\}, \forall i \in \{0\} \cup \{N\}, \forall j \in N, i \neq j, \ \forall k \in M \tag{10}$$

A função objetivo indicada em (1) é responsável pela minimização do *makespan*. O conjunto de restrições em (2) garante que cada tarefa, associada a apenas uma máquina, possui uma única tarefa imediatamente antecessora. O conjunto de restrições em (3) garante que existe, no máximo, uma tarefa imediatamente sucessora a outra, caso ambas sejam realizadas pela mesma máquina. O conjunto de restrições em (4) impõe que a tarefa fictícia 0, responsável pela inicialização do sistema, pode ser sucedida por apenas uma tarefa para cada máquina. Em (5), as restrições asseguram que os trabalhos predecessores e sucessores entre si são efetuados pela mesma máquina.

O conjunto de restrições em (6) certificam que o instante de término da tarefa j deve ocorrer após a soma entre o instante

de término da tarefa i (imediatamente anterior a j), o tempo de *setup* entre as tarefas i e j e o tempo de processamento de j. O conjunto de restrições em (7) estabelece que o instante de término da tarefa fictícia 0 é nulo para todas as máquinas. Em (8), o tempo total de conclusão das tarefas deve ser maior ou igual ao instante de término de cada tarefa para todas as máquinas. E, por fim, os conjuntos definidos em (9) e (10) indicam o domínio das variáveis de decisão.

O *Lower Bound* (limitante inferior) auxilia a busca de uma solução ótima de um problema [23]. Deste modo, segundo [23], o cálculo do *Lower Bound* ($\mathcal{L}$) para problemas com n tarefas a serem sequenciadas e m máquinas disponíveis, com o objetivo de minimizar o *makespan*, pode ser realizado da seguinte forma:

$$\mathcal{L} = \frac{1}{m}\left\{\sum_{i=1}^{n}\left[p_i + \min_{j\in\{1,..,n\}}(s_{ij})\right]\right\} \quad (11)$$

III. O MÉTODO LOCAL BRANCHING

Além da resolução do problema gerado por meio do modelo de PLIM apresentado em seção anterior, aplicou-se o método *Local Branching*. Este método consiste em realizar cortes no espaço de busca de soluções para, a partir de uma solução inicial, selecionar subespaços de solução adequados e explorá-los.

A. Solução inicial – Longest Expected Processing with Setup Time

Utilizou-se a heurística construtiva *Longest Expected Processing with Setup Time* (LEPST), proposta por [8], para inicializar o processo de resolução do problema abordado.

Seja S o conjunto de tarefas a serem sequenciadas. A heurística LEPST procede da seguinte forma:

(i) Encontrar a tarefa $j \in S$ tal que (12) possui o valor máximo

$$(p_j \times \alpha) + (1-\alpha) \times S_{M_j}; \quad (12)$$

onde S_{M_j} é o tempo de *setup* máximo da tarefa j quando sequenciado em qualquer máquina do conjunto M, ou seja, $S_{M_j} = \max\{s_{I_k,j}\}$. I_k é a última tarefa efetuada pela máquina k e o parâmetro α é encontrado no intervalo $[0,1]$. Quanto mais próximo o parâmetro α está de 0, maior é a consideração em relação ao tempo de *setup* para a escolha da tarefa inserida no sequenciamento. Por sua vez, o parâmetro α próximo a 1 indica a relevância dada ao tempo de processamento para selecionar a tarefa.

(ii) Remover a tarefa j do conjunto S e designá-la a máquina k de modo que (13) obtenha o valor mínimo

$$C_{I_k} + s_{I_k,j}; \quad (13)$$

onde C_{I_k} representa o término de processamento da última tarefa efetuada I_k.

(iii) Se S não possui tarefas a serem sequenciadas, então o processo é finalizado. Caso contrário, retorna-se ao Passo 1.

A heurística LEPST tem por objetivo inserir as tarefas com maior tempo de *setup* no início do sequenciamento, para que as tarefas com tempo de *setup* elevado não influenciem no final do sequenciamento, visando a minimização do *makespan*.

B. Local Branching

O *Local Branching* (LB) é um método exato de exploração parcial ou completa de vizinhanças, aplicado a problemas de PLIM. O método proposto por [22] busca explorar vizinhanças, a partir de uma solução inicial, por meio de ramificações (restrições de *Local Branching*) acrescidas ao problema de PLIM.

Para a aplicação do método, particiona-se o conjunto N, composto pelos índices das variáveis do problema, nos conjuntos de variáveis binárias B (não-vazio), inteiras G e contínuas C. O conjunto $\bar{S} = \{j \in B: \bar{x}_j = 1\}$ indica as variáveis binárias não nulas da solução incumbente $\bar{x}$.

A partir da solução incumbente $\bar{x}$, o espaço de solução associado ao nó de ramificação atual pode ser particionado, gerando uma ramificação a "esquerda" (14) e a "direita" (15), da seguinte forma:

$$\Delta(x,\bar{x}): \sum_{j\in\bar{S}}(1-x_j) + \sum_{j\in B\setminus\bar{S}} x_j \le k \quad (14)$$

$$\Delta(x,\bar{x}): \sum_{j\in\bar{S}}(1-x_j) + \sum_{j\in B\setminus\bar{S}} x_j \ge k+1 \quad (15)$$

Para um dado parâmetro k inteiro positivo, um subespaço contendo o conjunto de soluções viáveis do problema que satisfazem a restrição (14) define uma vizinhança $N(\bar{x},k)$ de $\bar{x}$.

Na restrição (15) são contabilizadas as variáveis binárias que eram unitárias na solução anterior e se tornaram nulas, assim como as variáveis que eram nulas e se tornaram unitárias. Ou seja, o número de trocas na quantidade de variáveis binárias do modelo deve ser menor ou igual ao parâmetro de vizinhança k. No trabalho de [22] sugere-se que este valor varie no intervalo de [10,20].

A cada nova ramificação há uma alteração do conjunto $\bar{S}$, acarretando na inserção de uma nova restrição (14) ao subproblema anterior, que por sua vez gerará uma nova vizinhança $N(\bar{x},k)$. Este processo é realizado até que não haja melhoria na solução corrente. Neste caso, realiza-se a busca no complementar da região já explorada.

IV. RESULTADOS E DISCUSSÕES

A abordagem apresentada na seção III e o modelo matemático de PLIM foram implementados na linguagem de programação VB.Net 2015 e para a resolução dos problemas utilizou-se o software de otimização *Gurobi Optimization* versão 7.0.1.

Os testes foram realizados em um computador com sistema operacional 64 Bits, *Microsoft Windows 7 Professional*, processador AMD A10 PRO-7800B R7, 12 Compute Core 4C + 8G, 3.50 GHz e 8 GB de memória RAM.

A. Testes Computacionais

As instâncias foram geradas de acordo com o trabalho de [7] e, além disso, realizou-se o cálculo do Limitante Inferior (ou *Lower Bound*) para todos os conjuntos de testes gerados, de acordo com a expressão (11), apresentada anteriormente.

Tanto para a abordagem de PLIM quanto para o *Local Branching* utilizou-se como critério de parada o tempo de 1 hora para cada problema. Os conjuntos de problemas gerados se dividem em 5 grupos com diferentes tempos de *setup* que variam de acordo com os tempos de processamento das tarefas (p_i e p_j). A seguir, apresentam-se os intervalos de geração dos grupos:

(i) *Grupo 1*: $[0{,}01 \times min(p_i, p_j); 0{,}1 \times min(p_i, p_j)]$;
(ii) *Grupo 2*: $[0{,}05 \times min(p_i, p_j); 0{,}1 \times min(p_i, p_j)]$;
(iii) *Grupo 3*: $[0{,}1 \times min(p_i, p_j); 0{,}2 \times min(p_i, p_j)]$;
(iv) *Grupo 4*: $[0{,}1 \times min(p_i, p_j); 0{,}5 \times min(p_i, p_j)]$;
(v) *Grupo 5*: $[0{,}2 \times min(p_i, p_j); 0{,}5 \times min(p_i, p_j)]$.

Ainda, os problemas variam quanto ao número de máquinas e tarefas. Foram testados problemas para 20, 30, 40, 50 e 100 tarefas, variando os números de máquinas disponíveis para 3 e 5 máquinas paralelas idênticas. Testou-se 30 replicações para cada conjunto gerado, totalizando 3000 testes de uma hora cada.

O resultado médio da Função Objetivo (Z) encontrado em 1 hora de teste foi utilizado para calcular o GAP (%) de acordo com a expressão (16):

$$GAP(\%) = \frac{Z - \mathcal{L}}{\mathcal{L}} \times 100 \qquad (16)$$

B. Testes Computacionais

Na Tabela I apresentam-se a média ($\bar{x}$) e o desvio padrão (s) para as 30 replicações dos valores de GAP obtidos para cada conjunto de teste. Para fins de comparação, apresentam-se os resultados para cada método separadamente, de acordo com os cinco grupos de *setup* (G1, G2, G3, G4 e G5), quantidade de máquinas e tarefas. Além disso, alguns resultados encontram-se em negrito, de modo a destacar os casos em que o método *Local Branching* apresenta os melhores resultados para ambas as estatísticas.

De modo geral, como pode ser observado na Tabela I, é possível verificar que, para os problemas gerados com 20 tarefas, tanto para 3 quanto para 5 máquinas paralelas, a abordagem PLIM apresentou resultados melhores, uma vez que o GAP é menor para todos os grupos de *setup* quando comparado aos valores obtidos pelo LB.

Paras os problemas que podem ser considerados de maior porte, ou seja, com número de tarefas a partir de 30 e dispondo de 5 máquinas paralelas no ambiente produtivo, o método LB apresenta resultados melhores que o PLIM. Estes resultados podem ser verificados tanto pelos valores de média quanto de desvio padrão obtidos de acordo com as replicações realizadas, que se encontram em destaque na Tabela I.

Já para o caso em que se dispõe de 3 máquinas, a abordagem PLIM apresenta valores médios de GAP menores, exceto para o caso de 100 tarefas no grupo 1, como encontra-se em destaque. Ainda, o aumento do número de máquinas mostra que a aplicação do LB apresenta melhoria significativa, como pode-se observar na última linha da Tabela I. Para os casos em que se tem o maior número de tarefas também encontram-se as maiores diferenças de resultados obtidos pelos dois métodos. Por exemplo, no Grupo 2, PLIM apresenta média de 16,22% de GAP e desvio padrão de 7,39%, enquanto o LB obteve GAP médio de 5,38% e um desvio de 0,19%.

Portanto, é possível afirmar, de acordo com as médias e desvios, que na medida em que o número de tarefas e o impacto do *setup* dos problemas aumenta, a viabilidade de aplicação do LB também aumenta, uma vez que os valores de desvio padrão e média diminuem.

Na Tabela II apresentam-se os intervalos de confiança referentes às médias apresentadas na Tabela I. Os Limites inferiores (LI) e superiores (LS) dos intervalos de confiança foram calculados conforme as expressões 17 e 18. Para as duas estratégias de solução adotou-se um nível de significância de 5%.

TABELA I
MÉDIA E DESVIO PADRÃO DO GAP (%)

		PLIM										***Local Branching***									
		G1		**G2**		**G3**		**G4**		**G5**		**G1**		**G2**		**G3**		**G4**		**G5**	
	Num Tarefas	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s	$\bar{x}$	s
3 Máquinas	20	1,04	0,28	1,98	0,42	3,65	1,07	5,24	1,55	7,53	2,52	1,52	0,38	2,53	1,04	4,9	1,31	8,28	1,76	10,42	2,57
	30	1,52	0,34	2,7	0,42	5,25	0,9	8,27	1,29	11,43	2,4	2,1	0,36	3,62	0,5	7,12	0,88	12,03	1,88	16,04	2,02
	40	2,13	0,29	3,47	0,45	6,51	1	10,34	1,6	13,25	2,06	2,48	0,4	4,24	0,35	8,47	0,8	14,94	1,73	19,03	1,81
	50	2,36	0,44	3,7	0,49	7,03	0,73	11,61	1,66	15,07	1,89	2,8	0,31	4,66	0,31	9,52	0,62	17,23	1,82	21,56	2,06
	100	4,17	2,06	5,55	2,1	10,33	2,94	18,5	2,38	21,19	1,87	3,43	0,33	5,63	0,24	11,55	0,48	22,12	1,56	27,1	1,4
5 Máquinas	20	2	0,75	2,77	0,85	4,66	1,15	6,59	1,95	8,1	2,23	2,21	0,46	3,29	1,28	5,83	1,38	7,92	1,65	9,69	2,59
	30	4,75	1,48	5,49	1,51	7,82	1,45	13,66	2,2	15,37	2,85	**2,29**	**0,42**	**3,57**	**0,63**	**6,87**	**0,91**	**11,19**	**2,12**	**14,82**	**1,74**
	40	7,3	2,65	8,88	2,58	12,22	2,92	16,87	3,69	20,07	3,63	**2,52**	**0,47**	**4,2**	**0,41**	**8,11**	**0,79**	**13,75**	**1,78**	**17,74**	**1,85**
	50	8,22	3,79	8,37	2,57	13,11	3,28	20,79	5,05	23,73	5,74	**2,68**	**0,33**	**4,68**	**0,4**	**9,08**	**0,61**	**13,98**	**1,61**	**19,88**	**2,01**
	100	11,31	5,74	16,22	7,39	20,44	7,51	30,15	8,16	33,17	6,85	**3,01**	**0,21**	**5,38**	**0,19**	**11,02**	**0,39**	**20,49**	**1,53**	**25,8**	**1,32**

$$LI = \bar{x} - t_{n-1,\alpha/2} \cdot \frac{s}{\sqrt{n}} \quad (17)$$

$$LS = \bar{x} + t_{n-1,\alpha/2} \cdot \frac{s}{\sqrt{n}} \quad (18)$$

De acordo com o intervalo de confiança é possível comparar os resultados obtidos por cada método, por exemplo, se há intersecção ou não entre os intervalos para o método PLIM e para o *Local Branching* em determinado problema para o respectivo grupo de *setup* testado. Os resultados que se encontram em negrito na Tabela II evidenciam os casos que além do método em destaque apresentar melhores resultados nas médias e desvios do GAP, não possuem intersecção entre nos intervalos de confiança. Sendo assim, é possível afirmar que quando se tem problemas com três máquinas e até 50 tarefas a solução do problema pelo PLIM é mais viável, já quando se dispõe de cinco máquinas e a partir de 30 tarefas a serem processadas, é mais satisfatória a aplicação do método *Local Branching*.

Para os casos que não há resultados em destaque na Tabela II para nenhum método, não foi possível afirmar que não há intersecção entre as médias, também não se pode apontar qual o melhor método a ser aplicado. Isto ocorre nos intervalos de confiança obtidos para os grupos G1, G2 e G3 quando se tem 100 tarefas e 3 máquinas; bem como G1, G2, G4 e G5 quando se tem um cenário com 20 tarefas e 5 máquinas e G5 com 30 tarefas e 5 máquinas. Sendo assim, aplicou-se o Teste t de diferença entre duas médias pareadas para comparar as médias dos valores de GAP obtidos pelos dois métodos aplicados.

O teste foi realizado comparando os resultados obtidos para as 30 replicações para cada problema, tanto para o método PLIM quanto para o *Local Branching*. Realizou-se o teste bilateral com nível de significância de 5% ($\propto = 0{,}05$). Considerou-se a hipótese nula H_0 de que não há diferença entre as médias obtidas e a hipótese alternativa H_1de que as médias obtidas são diferentes. Os resultados obtidos para os valores de p apresentam-se na Tabela III. Ainda, podem ser tomadas as seguintes decisões para os testes: i) $p > \propto$: Não há evidências para rejeitar a H_0; ii) $p \leq \propto$: Rejeita-se H_0.

TABELA III
RESULTADOS OBTIDOS PELO TESTE T

		G1	G2	G3	G4	G5
		p-valor				
3 Máquinas	20	8,53E-07	1,25E-05	2,92E-05	5,24E-09	4,27E-05
	30	4,66E-10	5,66E-13	1,92E-11	1,10E-14	9,76E-11
	40	3,87E-05	9,73E-11	9,37E-12	1,21E-16	1,05E-17
	50	1,42E-05	5,54E-14	2,09E-21	8,18E-19	8,00E-20
	100	**0,0057**	**0,0852**	2,72E-02	3,65E-10	3,99E-21
5 Máquinas	20	**0,1983**	**0,0688**	0,0007	0,0059	0,0133
	30	3,49E-12	2,95E-08	3,56E-03	4,22E-05	**0,3721**
	40	8,72E-14	6,01E-14	4,85E-10	1,04E-04	2,66E-03
	50	6,21E-11	1,50E-10	1,27E-08	1,61E-06	1,01E-03
	100	8,32E-11	5,52E-11	5,07E-09	3,32E-08	3,12E-07

De acordo com os valores de p obtidos, pode-se verificar que para os casos supracitados (que não foram destacados na Tabela II), é possível afirmar se as médias obtidas são significativamente diferentes ou não. Por exemplo, para G3 quando se tem 100 tarefas e 3 máquinas e para G4 e G5 com 20 tarefas e 5 máquinas, de acordo com os valores de p obtidos, a hipótese nula foi rejeitada.

Ou seja, podemos afirmar que há diferença entre as médias obtidas pelos métodos aplicados. Já para G1 e G2 com 20 tarefas, G5 com 30 tarefas e ambos com 5 máquinas e para G1 e G2 com 100 tarefas e 3 máquinas, verifica-se que não há evidências para rejeitar a hipótese nula, não se pode afirmar que há diferença significativa entre aplicar o método PLIM ou *Local Branching* para esses casos.

TABELA II
INTERVALOS DE CONFIANÇA

		PLIM										***Local Branching***									
		G1		**G2**		**G3**		**G4**		**G5**		**G1**		**G2**		**G3**		**G4**		**G5**	
		LI	LS	LI	LS	LI	LS	LI	LS	LI	LS	LI	LS	LI	LS	LI	LS	LI	LS	LI	LS
3 Máquinas	20	**0,94**	**1,14**	**1,82**	**2,14**	**3,25**	**4,05**	**4,66**	**5,82**	**6,59**	**8,47**	1,38	1,66	2,14	2,92	4,41	5,39	7,62	8,94	9,46	11,38
	30	**1,39**	**1,65**	**2,54**	**2,86**	**4,91**	**5,59**	**7,79**	**8,75**	**10,53**	**12,33**	1,97	2,23	3,43	3,81	6,79	7,45	11,33	12,73	15,29	16,79
	40	**2,02**	**2,24**	**3,30**	**3,64**	**6,14**	**6,88**	**9,74**	**10,94**	**12,48**	**14,02**	2,33	2,63	4,11	4,37	8,17	8,77	14,29	15,59	18,35	19,71
	50	**2,20**	**2,52**	**3,52**	**3,88**	**6,76**	**7,30**	**10,99**	**12,23**	**14,36**	**15,78**	2,68	2,92	4,54	4,78	9,29	9,75	16,55	17,91	20,79	22,33
	100	3,40	4,94	4,77	6,33	9,23	11,43	**17,61**	**19,39**	**20,49**	**21,89**	3,31	3,55	5,54	5,72	11,37	11,73	21,54	22,70	26,58	27,62
5 Máquinas	20	1,72	2,28	2,45	3,09	**4,23**	**5,09**	5,86	7,32	7,27	8,93	2,04	2,38	2,81	3,77	5,31	6,35	7,30	8,54	8,72	10,66
	30	4,20	5,30	4,93	6,05	7,28	8,36	12,84	14,48	14,31	16,43	**2,13**	**2,45**	**3,33**	**3,81**	**6,53**	**7,21**	**10,40**	**11,98**	14,17	15,47
	40	6,31	8,29	7,92	9,84	11,13	13,31	15,49	18,25	18,71	21,43	**2,34**	**2,70**	**4,05**	**4,35**	**7,82**	**8,40**	**13,09**	**14,41**	**17,05**	**18,43**
	50	6,80	9,64	7,41	9,33	11,89	14,33	18,90	22,68	21,59	25,87	**2,56**	**2,80**	**4,53**	**4,83**	**8,85**	**9,31**	**13,38**	**14,58**	**19,13**	**20,63**
	100	9,17	13,45	13,46	18,98	17,64	23,24	27,10	33,20	30,61	35,73	**2,93**	**3,09**	**5,31**	**5,45**	**10,87**	**11,17**	**19,92**	**21,06**	**25,31**	**26,29**

Ainda, para os demais problemas, de acordo com os resultados de *p* obtidos pelo teste, rejeitou-se H_0, o que mostra que as médias obtidas pelos dois métodos são significativamente diferentes.

Observa-se também a influência do tempo de *setup* nos resultados. Nota-se que, em todas as variações dos conjuntos, quanto maior o tempo de *setup*, maior é o valor do GAP obtido. Apresenta-se na Figura 1 uma análise dos valores obtidos para o GAP para os dois métodos aplicados, variando o número de máquinas (m=3 e m=5). Os gráficos apresentam as curvas dos resultados obtidos variando os números de tarefas para cada grupo de *setup*.

Como pode-se observar na Figura 1, o grupo com menor tempo de *setup* (Grupo 1), no caso de 3 máquinas paralelas é o único grupo em que o método LB alcança melhores resultados quanto ao GAP se comparado ao PLIM para 100 tarefas. Já para este mesmo caso, quando o número de tarefas varia entre 20 e 50 a abordagem PLIM apresenta melhores resultados em relação ao GAP obtido. Além disso, no decorrer dos grupos a diferença de GAP entre as duas abordagens se acentua, ou seja, quanto maior o tempo de *setup*, a aplicação da abordagem PLIM se mostra mais vantajosa de acordo com o critério de parada.

É possível verificar de acordo com os gráficos apresentados que, de modo geral, quanto maior o número de tarefas e máquinas a abordagem LB se mostra mais robusta. Uma vez que, quanto maiores os problemas tratados, o LB apresenta resultados mais próximos aos da solução ótima em tempo computacional viável.

V. CONCLUSÕES

Neste trabalho foram propostas duas abordagens de solução para o problema de sequenciamento em máquinas paralelas idênticas, com o objetivo de minimização do *makespan* e *setup* dependente da sequência, um modelo de Programação Linear Inteira Mista e o *Local Branching*. Foram realizados 3000 testes computacionais de 1 hora cada, variando-se o número de tarefas, o número máquinas e os tempos de *setup*. Além disso, calculou-se o Limitante Inferior *(Lower Bound*) dos problemas para fins de comparação de resultados quanto ao GAP para as instâncias testadas.

Os resultados obtidos mostraram que para problemas maiores, ou seja, quando se tem 30 ou mais tarefas e 5 máquinas, o método *Local Branching* apresenta melhores soluções que a abordagem PLIM para o critério de parada utilizado. No entanto, de modo geral, quando se tem 3 máquinas a PLIM apresenta resultados melhores.

Foi possível verificar também que quanto maior o número de tarefas e máquinas a abordagem LB apresenta maior robustez, visto que, de acordo com os resultados foi possível constatar que enquanto o *Local Branching* se aproxima das soluções ótimas para estes casos, a abordagem PLIM se torna cada vez menos viável.

Como propostas de trabalhos futuros, sugere-se a aplicação do método *Local Branching* para demais ambientes de produção, bem como para diversas características apresentadas pelos problemas de produção. Ademais, visto que o método aplicado se mostrou uma abordagem de solução promissora para o SMPI, principalmente para problemas de grande porte, sugere-se também, um refinamento desta estratégia de solução. Como por exemplo, a intensificação da busca local no método, fazendo uso de heurísticas ou meta heurísticas durante a solução em determinada ramificação.

AGRADECIMENTOS

Os autores agradecem à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro.

REFERÊNCIAS

[1] G. A. Schneider, F. N. Jr, L. Magatão, and L. V. R. Arruda, "A Mathematical Programming Approach to Optimize the Scheduling of Tanks in Oil Refineries," IEEE Lat. Am. Trans., vol. 14, no. 2, pp. 818–830, 2016.

[2] F. A. P. Pinto, L. G. D. M. Leite, G. C. Barroso, and M. F. Aguilar, "Algorithms Scheduling with Migration Strategies for Reducing Fragmentation in Distributed Systems," IEEE Lat. Am. Trans., vol. 13, no. 3, pp. 762–768, 2015.

[3] J. Y. T. Leung, K. Lee, and M. L. Pinedo, "Bi-criteria scheduling with machine assignment costs," *Int. J. Prod. Econ.*, vol. 139, no. 1, pp. 321–329, 2012.

[4] A. Allahverdi, H. Aydilek, and A. Aydilek, "Single machine scheduling problem with interval processing times to minimize mean weighted completion time," *Comput. Oper. Res.*, vol. 51, pp. 200–207, 2014.

[5] E. Vallada and R. Ruiz, "A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times," *Eur. J. Oper. Res.*, vol. 211, no. 3, pp. 612–622, 2011.

[6] M. L. Pinedo, *Scheduling - Theory, Algorithms, and Systems*. 2008.

[7] D. Nait Tahar, F. Yalaoui, C. Chu, and L. Amodeo, "A linear programming approach for identical parallel machine scheduling with job splitting and sequence-dependent setup times," *Int. J. Prod. Econ.*, vol. 99, no. 1–2, pp. 63–73, 2006.

[8] J.-P. Arnaout, "Heuristics for the Maximization of Operating Rooms Utilization Using Simulation," *Simulation*, vol. 86, no. 8–9, pp. 573–583, Aug. 2010.

[9] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. R. Kan, "Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey," in *Discrete Optimization II*, vol. 5, no. Supplement C, P. L. Hammer, E. L. Johnson, and B. H. Korte, Eds. Elsevier, 1979, pp. 287–326.

[10] A. Allahverdi and H. M. Soroush, "The significance of reducing setup times/setup costs," *Eur. J. Oper. Res.*, vol. 187, no. 3, pp. 978–984, 2008.

[11] M. A. N. D. G.-S. F. MONTOYA-TORRES JAIRO R. AND SOTO-FERRARI, "PRODUCTION SCHEDULING WITH SEQUENCE-DEPENDENT SETUPS AND JOB RELEASE TIMES," *DYNA*, vol. 77, pp. 260–269, Nov. 2010.

[12] F. Yalaoui and C. Chu, "An efficient heuristic approach for parallel machine scheduling with job splitting and sequence-dependent setup times," *IIE Trans.*, vol. 35, no. 2, pp. 183–190, 2003.

[13] X. Li, F. Yalaoui, L. Amodeo, and H. Chehade, "Metaheuristics and exact methods to solve a multiobjective parallel machines scheduling problem," *J. Intell. Manuf.*, vol. 23, no. 4, pp. 1179–1194, Aug. 2012.

[14] L. R. Abreu and B. A. Prata, "A hybrid genetic algorithm for solving the unrelated parallel machine scheduling problem with sequence dependent setup times," *IEEE Lat. Am. Trans.*, vol. 16, no. 6, pp. 1715–1722, 2018.

[15] A. S. Mendes, F. M. Müller, P. M. França, and P. Moscato, "Comparing meta-heuristic approaches for parallel machine scheduling problems," *Prod. Plan. Control*, vol. 13, no. 2, pp. 143–154, 2002.

[16] A. Hamzadayi and G. Yildiz, "Modeling and Solving Static M Identical Parallel Machines Scheduling Problem with a Common Server and Sequence Dependent Setup Times," *Comput. Ind. Eng.*, vol. 106, no. C, pp. 287–298, Apr. 2017.

[17] B.-K. Kim and Y.-D. Kim, "Heuristic algorithms for assigning and scheduling flight missions in a military aviation unit," *Comput. Ind. Eng.*, vol. 61, no. 4, pp. 1309–1317, 2011.

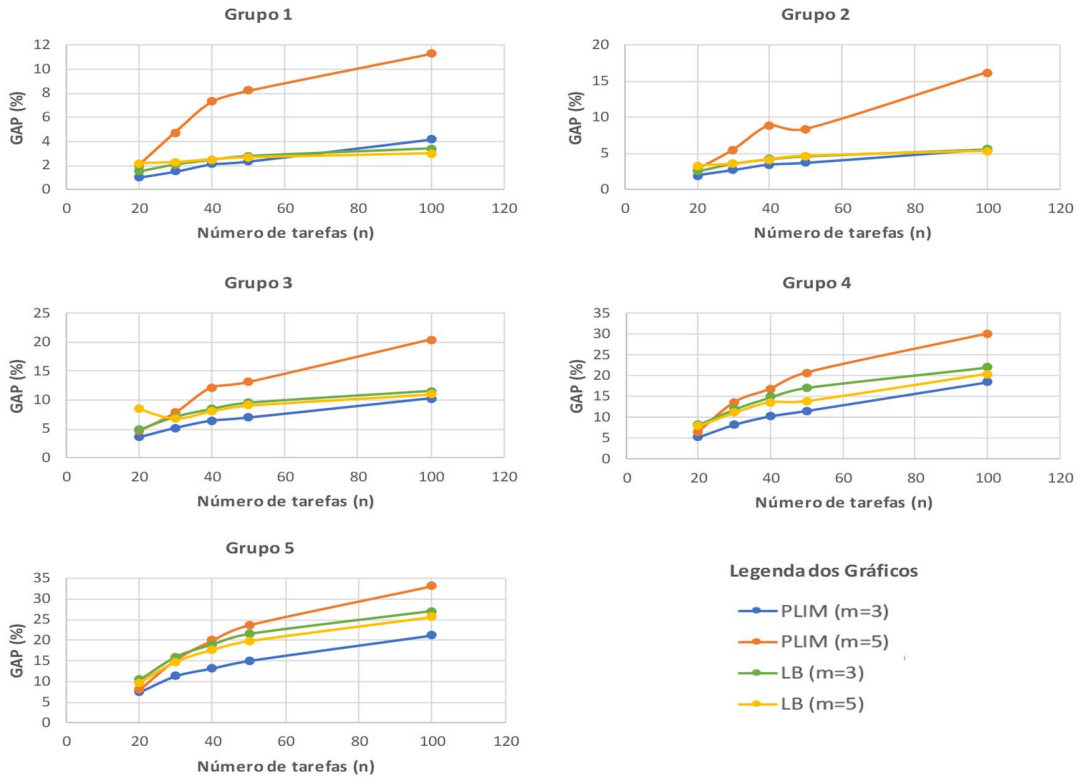

Figura 1. Resultados em relação ao número de tarefas de acordo com os grupos de tempo de *setup*.

[18] C.-H. Lee, C.-J. Liao, and T.-P. Chung, "Scheduling with multi-attribute setup times on two identical parallel machines," Int. J. Prod. Econ., vol. 153, no. Supplement C, pp. 130–138, 2014.

[19] A. D. Wilson, R. E. King, and T. J. Hodgson, "Scheduling non-similar groups on a flow line: multiple group setups," *Robot. Comput. Integr. Manuf.*, vol. 20, no. 6, pp. 505–515, 2004.

[20] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[21] M. Samavati, D. Essam, M. Nehring, and R. Sarker, "A local branching heuristic for the open pit mine production scheduling problem," *Eur. J. Oper. Res.*, vol. 257, pp. 261–271, 2017.

[22] M. Fischetti and A. Lodi, "Local branching," *Math. Program.*, vol. 98, no. 1–3, pp. 23–47, 2003.

[23] M. E. Kurz and R. G. Askin, "Heuristic scheduling of parallel machines with sequence-dependent set-up times," *Int. J. Prod. Res.*, vol. 39, no. 16, pp. 3747–3769, 2001.

Talita Mariana Pinho Schimidt Engenheira Química (2014) pela Universidade Estadual do Oeste do Paraná (UNIOESTE). Mestre em Engenharia de Produção (2017) na área de concentração de Pesquisa Operacional pelo Programa de Pós Graduação em Engenharia de Produção (PPGEP) da Universidade Federal do Paraná (UFPR) e atualmente cursa Doutorado no Programa de Pós Graduação em Métodos Numéricos em Engenharia pela mesma Universidade. É aluna do Grupo de Tecnologia Aplicada à Otimização (GTAO). Atualmente suas pesquisas se concentram nas áreas: Pesquisa Operacional, Planejamento e Controle da Produção (PCP) e Programação Inteira Mista (PIM).

Cassius Tadeu Scarpin Doutor (2012) e Mestre (2007) em Pesquisa Operacional pelo Programa de Pós Graduação em Métodos Numéricos em Engenharia, área de concentração: Programação Matemática, na Universidade Federal do Paraná. Engenheiro de Produção (2010) e Licenciado em Matemática (2002) pela mesma universidade. Atualmente ocupa o cargo de Professor Adjunto no Departamento Administração Geral e Aplicada (DAGA) da de Produção da Universidade Federal do Paraná (UFPR). Possui experiência na área de Engenharia de Produção, Gestão de Operações e Logísitica, com particular interesse em Pesquisa Operacional e Logística, atuando principalmente nos seguintes temas: Sistema de Reposição de Estoques, Engenharia da Qualidade, Mapeamento e Análise de Processos, Hierarquia/Regionalização, Carregamento e Roteamento de Veículos, Otimização em fluxo de Pacientes, Previsão de Séries Temporais no Varejo, Metaheurísticas e Otimização na Separação de Produtos.

Gustavo Valentim Loch Possui graduação em Matemática Industrial pela Universidade Federal do Paraná (2007), graduação em Ciências Contábeis pela Universidade Positivo (2011), mestrado em Métodos Numéricos em Engenharia pela Universidade Federal do Paraná (2010) e doutorado em Métodos Numéricos em Engenharia pela Universidade Federal do Paraná (2014). Atualmente é professor do Departamento de Administração Geral e Aplicada da Universidade Federal do Paraná, atuando principalmente nos seguintes temas: Otimização combinatória, Problema de Transporte, Engenharia da Qualidade e métodos de auxilio à decisão. Em 2015 foi premiado com Menção Honrosa no Prêmio Capes de Tese 2015.

Nathália Cristina Ortiz da Silva possui graduação em Matemática Bacharelado pela Universidade Estadual de Maringá (2009) e mestrado em Métodos Numéricos em Engenharia pela Universidade Federal do Paraná (2016). Atualmente é professora efetiva do Instituto Federal do Rio Grande do Sul. Tem experiência na área de Pesquisa Operacional, com ênfase em Programação da Produção, atuando principalmente nos seguintes temas: Sequenciamento de Produção e Simulação.