Revista Latino-Americana de Inovação e Engenharia de Produção

APLICAÇÃO DE UMA NOVA HEURÍSTICA EM PROBLEMAS DE *JOB SHOP* COM TEMPO VARIÁVEL DEPENDENTE DA SEQUÊNCIA

A NEW HEURISTIC APPLICATION IN JOB SHOP SCHEDULING PROBLEMS WITH SETUP TIME WITH SEQUENCE DEPENDENCE

Francisco Bartosievicz Netto [1]
Arinei Carlos Lindbeck da Silva [2]
Dani Juliano Czelusniak [3]

RESUMO: O Planejamento e Controle da Produção é uma área de estudo que pode, juntamente à Pesquisa Operacional e a modelagem matemática, otimizar o planejamento da produção. A programação da produção é ponto crucial na determinação de como sequenciar as tarefas pelas máquinas que as mesmas devem passar para então chegar ao produto final. No ambiente das oficinas de máquina pode-se considerar um possível tempo de preparação da máquina antes de receber a próxima tarefa e, este tempo depende da última tarefa sequenciada nesta máquina, o que torna o problema ainda maior e mais complexo. Neste contexto, este trabalho tem como objetivo aplicar uma heurística que busca minimizar o tempo total de trabalho, tendo sido obtidos nos testes, resultados satisfatórios.

Palavras-chave: *Job Shop*. *Scheduling*. Dependência da Sequência. Tempo de preparação.

1 INTRODUÇÃO

O problema de sequenciamento pode ser entendido, num primeiro momento, como o procedimento que se utiliza de regras de decisão lógica para selecionar uma ordem de produção, ou seja, que tarefa (objeto, produto, etc.), dentre as disponíveis para ser processada, será alocada a uma máquina que esteja livre para processar e em que momento este processamento deve ocorrer.

[1] Doutor, Pontifícia Universidade Católica do Paraná, Escola Politécnica, Curitiba/Brasil. francisco.bartosievi@pucpr.br

[2] Doutor, Universidade Federal do Paraná, Curso de Engenharia de Produção, Curitiba/Brasil. arinei@ufpr.br

[3] Doutor, Pontifícia Universidade Católica do Paraná, Escola Politécnica, Curitiba/Brasil. dani.czelusniak@pucpr.br

Conforme descrito por Colin (2007), as regras que estabelecem a forma que o sequenciamento deve ocorrer, pertencem aos primórdios da administração produtiva e ainda hoje podem ser consideradas um poderoso instrumental a ser usado no dia a dia. A teoria em si foi iniciada na primeira metade da década de 1950, com três trabalhos publicados entre 1954 e 1956 que influenciaram de maneira profunda todo o desenvolvimento teórico posterior. Nestes trabalhos, regras para a prioridade do sequenciamento, como *Shortest Processing Time* (SPT), *Weighted Shortest Processing Time* (WSPT) e *Earliest Due Date* (EDD), foram definidas e estudadas formalmente. As regras já eram conhecidas nos ambientes profissionais, mas até então não haviam sido estudadas formalmente.

Proud (1999) descreve que o problema de sequenciamento da produção tomou forma e começou a ser estudado de fato na década de 1960, após uma nova geração começar a se formar após a Segunda Grande Guerra Mundial. As companhias norte-americanas começaram a se preocupar com o uso otimizado de insumos na produção de seus produtos, com o objetivo de manufaturar apenas a quantidade necessária para a venda. Infelizmente, a manufatura raramente produzia apenas o que seria vendido. E com a invariável mudança de demanda, a quantidade dos insumos também mudava. Com o desenvolvimento de ferramentas computacionais, tornou-se possível gerir a esmagadora mudança de planejamento que as plantas de produção e os fornecedores até então não podiam lidar.

2 REVISÃO DE LITERATURA

Esta seção apresenta uma revisão da produção científica em torno do assunto. Os trabalhos pesquisados e estudados trouxeram diferentes contribuições para o problema principal em termos de novos conceitos, modelos e métodos de solução.

Rocha et al. (2004) utilizaram a meta-heurística GRASP para obter um limitante da execução do algoritmo de *Branch and Bound* para resolver o modelo proposto para o problema de máquinas paralelas com tempo de preparação dependente da sequência.

Alvarez-Valdes et al. (2005) consideraram uma situação particular de *job shop* aplicada a uma fábrica de produto de vidro, levando em conta as características das máquinas e das diferentes formas de processo dos produtos. Eles consideraram como objetivo custos e o prazo de entrega dos produtos aos clientes. Assim como neste trabalho e, em diversos outros pesquisados, os problemas de *job shop* muitas vezes são apresentados em situações de problemas específicos.

Nowicki e Smutnicki (2005) apresentaram um algoritmo baseado na "Busca Tabu" para apresentar soluções para o *job shop*, em uma situação genérica, com o *makespan* como objetivo, porém, não consideraram o tempo de preparação das máquinas. Liu et al. (2006) consideraram o mesmo problema, mas propuseram um algoritmo genético melhorado baseado no algoritmo híbrido *Taguchi-Genetic*, que busca uma melhoria entre as etapas de crossover e da mutação. Já Huang e Liao (2008) utilizaram um híbrido entre a meta-heurística de "Otimização por Colônia de Formigas" e "Busca Tabu".

Ruiz et al. (2005) consideraram o tempo de preparação das máquinas dependentes da sequência com o objetivo de otimizar o tempo total de trabalho, porém, no ambiente de *flow shop*, através de buscas locais de permutações das tarefas sequenciadas. Neste mesmo trabalho, eles citam que buscaram os melhores parâmetros para as meta-heurísticas utilizadas por delineamento de experimentos. Tasgetiren et al. (2007) trabalharam o mesmo problema, mas sem o tempo de preparação das máquinas, utilizando a meta-heurística de "Otimização por Núvem de Partículas".

Tanto Algoritmos Genéticos quanto o *Simulated Annealing* utilizam-se de parâmetros para sua execução, e Sadegheih (2006) apresentou um estudo sobre a influência dos mesmos sobre os problemas de *flow shop* e *job shop*, de forma comparativa entre as duas formas de solução. As conclusões foram sobre como a velocidade de resfriamento do sistema influência no *Simulated Annealing* e o quanto a amplitude de intervalo dos parâmetros de um Algoritmo Genético podem deteriorar (termo usado por Sadegheih) a qualidade das soluções, e ainda sobre o quanto os Algoritmos Genéticos são mais rápidos que o *Simulated Annealing*.

Monkman et al. (2008) apresentaram mais um trabalho em uma situação específica do problema de *job shop*, neste caso, uma indústria de produtos eletrônicos. Eles consideraram o tempo de preparação dependente da sequência, mas na forma de custo, e seu objetivo foi minimizar o custo com os tempos de preparação. Para tal, utilizaram-se de uma heurística que envolveu três etapas - designação, sequenciamento e tempo de programação (no conceito de *scheduling*) - através do GRASP.

Qian et al. (2008) consideraram o *job shop* de forma multi-objetivo levando em conta o tempo total de trabalho e o máximo atraso. Para resolver seu modelo, eles utilizaram a meta-heurística dos Algoritmos Meméticos baseados em Evolução Diferencial, tanto com busca global, quanto em busca adaptativa local.

Luo et al. (2009) apresentaram um problema em dois estágios de sequenciamento em *flow shop* por lotes em um caso específico de uma empresa de metalurgia. O primeiro estágio

tem quatro máquinas em paralelo e o segundo, apenas uma máquina. Eles consideraram itens específicos para a produção desta empresa, dentre os quais o tempo de preparação dependente da sequência. Para solucionar o modelo desenvolvido por eles, foi utilizado um Algoritmo Genético.

Manikas e Cheng (2009) trabalharam o problema de *job shop* de forma multi-critério no objetivo e realizaram testes com várias funções objetivo considerando ponderações entre o tempo total de trabalho, atraso máximo, entrega dos produtos, e resolveram os modelos por Algoritmos Genéticos. Eles citam a questão do tempo de processamento dependente da sequência no início do artigo, mas na descrição do modelo e nos exemplos executados, não fica claro se foi de fato considerado na resolução dos problemas citados, mais parecendo que não foram considerados, ou que os mesmos foram incorporados aos tempos de processamento das tarefas, muito embora este tempo seja dependente da tarefa executada anteriormente na máquina em questão.

Um problema de *job shop* com o objetivo de minimizar o atraso de entrega foi trabalhado por Zhou et al. (2009). Eles utilizaram um híbrido de Heurística e Algoritmo Genético determinando qual tarefa deve ser processada por primeiro em cada máquina e a Heurística a designação de qual o resto da sequência das tarefas nas máquinas. Eles não consideraram o tempo de preparação dependente da sequência em seu modelo.

Parthanadee e Buddhakulsomsiri (2010) realizaram simulações de modelos e analisaram os resultados para determinar qual a regra de prioridade mais apropriada para um estudo de caso específico na indústria de frutas (na Tailândia), levando em conta características do problema como a incerteza de haver frutas boas e na quantidade necessária disponíveis, diferentes produtos finais que compartilham o mesmo insumo tornando o sequenciamento das tarefas interdependentes entre si. O interessante neste artigo é que, em nenhum momento, eles citam se o problema apresentado por eles está no ambiente de *flow shop* ou *job shop*, apesar de citarem artigos na revisão da literatura que trabalham em um ou outro ambiente.

Mati et al. (2011) apresentaram um modelo de busca local e troca de vizinhança de arcos disjuntos no grafo que o problema de *scheduling* pode ser representado, no ambiente *job shop*, buscando otimizar qualquer critério indicador de avaliação da produção como os descritos na seção anterior. O modelo proposto é resolvido por uma heurística dependente de apenas um parâmetro, e não foi considerado o tempo de preparação das máquinas. Sels et al. (2011) também trabalharam o problema de *job shop*, mas através de um Algoritmo Genético e um procedimento de busca dispersa, trabalhando não apenas com uma população, mas com

duas populações, e assim, com uma maior diversidade e qualidade de informação, para então efetuar as trocas entre indivíduos mais bem classificados das distintas populações, no intuito de adicionar diversidade no processo de pesquisa de soluções. Eles também buscaram mostrar vantagens nesta forma de Algoritmo Genético sobre algumas meta-heurísticas.

Georgiadis e Michaloudis (2012) utilizaram um sistema dinâmico para representar o problema de *job shop*, examinando um caso específico de chão de fábrica, em que os efeitos da chegada de pedidos feitos pelos clientes e os vários tipos de eventos que podem ocorrer na situação real em função destes pedidos e, também, de falhas de máquinas são considerados. Apesar do sistema apresentado por eles buscar uma proximidade maior entre a teoria e a realidade, o tempo de preparação dependente da sequência não é utilizado, no sentido literal de preparação e ajustes na máquina antes da tarefa efetivamente ser realizada nas máquinas. Este apenas é citado como muitas vezes necessário e, no sistema dos autores é considerado apenas o tempo independente da sequência. O termo *setup* muitas vezes é empregado na execução em si da tarefa na máquina.

Shen e Buscher (2012) consideram o tempo de preparação da máquina dependente da sequência buscando minimizar o tempo total de trabalho no ambiente de *job shop*. Mas na intenção de otimizar o processo, é primeiro realizado o loteamento das tarefas, quase que como um agrupamento de tarefas com certo grau de semelhança, levando a uma diminuição considerável do número de tarefas efetivas a ser sequenciadas, para então aplicar um algoritmo do tipo "Busca Tabu" e chegar à solução.

Finalmente, mais recentemente, Karimi-Nasab e Seyedhoseini (2013) propuseram um modelo de Programação Linear Inteira para efetuar loteamento das tarefas e seu sequenciamento, no ambiente *job shop*, e consideraram o tempo de preparação dependente da sequência, mas na forma de otimizar o custo financeiro destes tempos de preparação e não com o objetivo de minimizar o tempo total de trabalho. Foi incorporado ao modelo a ideia de máquinas flexíveis que permitem o gerenciador da produção mudar a velocidade de trabalho das máquinas. Como o modelo tem diversas restrições de desigualdade, e estas reduzem o espaço de soluções, elas são tratadas com o que é chamado no trabalho de planos de corte (*cutting planes*). Estas são utilizadas para resolver o modelo em *cut and branch* e *branch and cut*. Eles executaram o modelo no software de simulação CPlex, da IBM. Os autores citam nos resultados que o modelo tenta ser bem próximo do problema real, mas que para ser aplicado em situações de muitas tarefas e máquinas (em situações reais de fato), deve-se previamente estabelecer os planos de corte de forma a passar em pontos específicos do espaço

de soluções, e que possivelmente outras formas de solução como decomposições ou relaxamento de Lagrange possam trazer melhores resultados no futuro.

3 METODOLOGIA DE PESQUISA

A heurística aplicada nos problemas de *job shop scheduling* que leva em conta a dependência da sequência das tarefas é a apresentada a seguir e o mesmo utiliza-se das seguintes notações:

i – tarefa

j – máquina

T_{ik} – Matriz tecnológica representando a tarefa i na etapa k da sua sequência tecnológica, ainda $T_{ik} = j$

O_{ij} – Operação da tarefa i a ser executada na máquina j

$TI(O_{ij})$ – Tempo de início da execução da tarefa i na máquina j

$TC(O_{ij})$ – Tempo de conclusão (término) da execução da tarefa i na máquina j

$TP(O_{ij})$ – Tempo de preparação (*setup*) para a tarefa i poder ser executada na máquina j

$TE(O_{ij})$ – Tempo de execução da tarefa i na máquina j

$MC(O_{ij})$ – Tempo de conclusão mais cedo da tarefa i na máquina j

Γ_j – Tempo de conclusão da última tarefa sequenciada na máquina j

Φ_j – Última tarefa sequenciada na máquina j

$\Sigma_{j,\Phi j,i}$ – Matriz de tempo de preparação da máquina j, com tarefa Φ_j, com a próxima tarefa a ser executada i

Ω – Lista com as tarefas sequenciadas pelo algoritmo

Os passos do algoritmo são:

- Passo 1: Iniciar G como um conjunto das operações que são as primeiras de cada tarefa na sequência tecnológica.

$$G = \{O_{1,T11}, O_{2,T21}, \ldots, O_{n,Tn1}\}$$

Para cada operação $O_{ij} \in G$, fazer $TI(O_{ij}):=0$. Iniciar S:={ }.

- Passo 2: Achar a operação que termina mais cedo no conjunto G, considerando o tempo de preparação necessário para a tarefa entrar na máquina, em função da última operação sequenciada na mesma.

$$O_{i*r} = \text{argmin}_i \{ MC(O_{ij}) \mid O_{ij} \in G \}$$

Onde:

$$MC(O_{ij}) = \begin{cases} \Gamma_j + TE(O_{ij}) + \Sigma_{j,\Phi j,i}, \text{ se } \Gamma_j \geq TI(O_{ij}) \\ \Gamma_j + TE(O_{ij}) + (\Gamma_j + \Sigma_{j,\Phi\_j,i} - TI(O_{ij})), \text{ se } \Gamma_j + \Sigma_{j,\Phi j,i} \geq TI(O_{ij}) \\ \Gamma_j + TE(O_{ij}), \text{ caso contrário.} \end{cases}$$

- Passo 3: Montar o conjunto conflito C, subconjunto de G, em que o tempo de início da operação $O_{ir} \in G$ mais o tempo de preparação necessário para executar a tarefa i na máquina r seja menor que o tempo de término da operação O_{i*r}.

$$C = \{ O_{ir} \in G \mid MC(O_{ir}) - TE(O_{ir}) \leq MC(O_{i*r}) \}$$

- Passo 4: Selecionar uma das operações do conjunto conflito C aleatoriamente, denominando esta operação como O_{xr}.

$$O_{xr} = \text{rnd} \{O_{ir} \in C\}$$

- Passo 5: Sequenciar O_{xr} na lista Ω que contém as operações já sequenciadas O_{ij}, com as informações de tempo de preparação $TP(O_{ij})$, tempo de conclusão $TC(O_{ij})$ e tempo de início $TI(O_{ij})$ da operação.

$$\Omega = \Omega \cup \{O_{xr}\}$$

Onde:

$TP(O_{xr}) = \Sigma_{r,\Phi r,k}$

$TC(O_{xr}) = MC(O_{kr})$

$TS(O_{xr}) = MC(O_{kr}) - TE(O_{kr})$

- Passo 6: Atualizar o tempo de todas as outras operações O_{ir} em G da forma:

$$TI(O_{ir}) = \max \{ TI(O_{ir}) ; MC(O_{xr}) \}, \forall O_{ir} \in (C - \{O_{xr}\})$$

- Passo 7: Remover a operação sequenciada O_{xr} de G e adicionar a próxima operação O_{xj} da sua sequência tecnológica (se ainda houver operações a sequenciar).

$$G = (G - \{O_{xr}\}) \cup \{O_{xj}\}$$

Atualizando sua informação de tempo inicial:

$$TI(O_{kj}) = \max\{ TC(O_{kr}) ; TC(\Phi_j)\}$$

- Passo 8: Repetir do Passo 2 ao Passo 7 até que todas as operações de todas as tarefas estejam sequenciadas.
- Passo 9: Apresentar a lista Ω como um sequenciamento.

Para validação dos resultados obtidos pela heurística aqui apresentada foi considerado o modelo de Programação Inteira Mista (PIM) apresentado a seguir. Tal modelo tem como limitação, a necessidade de que cada tarefa passe por todas as máquinas uma e apenas uma vez. Este modelo foi adaptado do apresentado em Cheung e Zhou (2001).

O modelo utiliza-se das seguintes notações:

- n - número de tarefas
- m - número de máquinas
- i - tarefa i, i = 1, 2, … , n
- j - máquina j, j = 1, 2, … , m
- M - constante positiva suficientemente grande

Parâmetros:

- p(*ji*) - tempo de processamento da tarefa i na máquina j; i = 1, 2, … , n; j = 1, 2, … , m
- σ^i - sequência tecnológica da tarefa i, isto é, a ordem das máquinas que a tarefa i deve passar, $\sigma^i = (\sigma^{i1}, \sigma^{i2}, \dots, \sigma^{im})$, $\sigma^{ij} \in \{1, 2, \dots, m\}$ é o índice da correspondente máquina.
- s^j_{ik} - tempo de preparação requerido para processar a tarefa k na máquina j imediatamente após a tarefa i ter sido processada na máquina j, j = 1, 2, … , m; i,k= 1, 2, … , n.

Variáveis:

- C_{max} - *makespan* da programação da produção
- t(ji) - tempo de início do processamento da tarefa i na máquina j
- $x^j_{ik} = \begin{cases} 1, \text{ se a tarefa k é sequenciada imediatamente após i na máquina j,} \\ 0, \text{ caso contrário} \end{cases}$

Modelo:

Função Objetivo: min Cmax

Sujeito a:

(1) $t(\sigma^i_h i) + p(\sigma^i_h i) \leq t(\sigma^i_{h+1} i)$, $h = 1, 2, \dots, m-1$; $i = 1, 2, \dots, n$

(2) $t(\sigma^i_m i) + p(\sigma^i_m i) \leq C_{max}$, $i = 1, 2, \dots, n$

(3) $t(ji) + p(ji) + s^j_{ik} x^j_{ik} \leq t(jk) + (1-x^j_{ik}) M$, $j = 1, 2, \dots, m$; $i = 1, \dots, n$, $k = 1, 2, \dots, n$, $k \neq i$

(4) $\Sigma x^j_{ik} = 1$, para $k=0$ $(k \neq i)$ até n, $j = 1, 2, \dots, m$; $i = 0, 1, \dots, n$

(5) $\Sigma x^j_{ik} = 1$, para $i=0$ $(i \neq k)$ até n, $j = 1, 2, \dots, m$; $k = 0, 1, \dots, n$

(6) $x^j_{ik} + x^j_{ki} \leq 1$, $j = 1, 2, \dots, m$; $i,k = 0, 1, 2, \dots, n$, $i \neq k$

Sobre as restrições:

- (1) garante que a sequência tecnológica de cada tarefa seja respeitada,
- (2) é uma restrição natural em que todas as operações devem ser completas antes do *makespan*,
- (3) garante que a máquina só pode começar a processar a próxima tarefa apenas quando a operação prévia e o tempo de *setup* estejam completos,
- (4) na máquina j quando uma tarefa complete seu processamento, uma e apenas uma tarefa será selecionada para ser sequenciada após esta, a menos que a máquina j complete todas as operações,
- (5) implica em que a operação de uma tarefa deve ter uma e apenas uma tarefa predecessora, exceto quando a mesma é a primeira operação, e
- (6) garante que uma tarefa não seja sequenciada após outra que seja sua antecessora.

Este modelo tem $(m-1)n + n + mn(n-1) + 2m(n+1) + (mn(n+1))/(2)$ restrições, mn variáveis positivas t(ji) e $mn(n+1)$ variáveis binárias xjik. Assim, para um problema 10x10, por exemplo, o modelo terá 1770 restrições, 100 variáveis positivas e 1100 variáveis binárias, o que torna este modelo de difícil execução para problemas grandes.

4 RESULTADOS E CONSIDERAÇÕES FINAIS

Nesta seção são apresentados os testes realizados com o programa desenvolvido sobre a heurística apresentada. O computador utilizado foi um notebook HP Pavilion dm4-1075br, com processador Intel(R) Core(TM) i5 CPU 2.27GHz, 4,00GB de memória RAM. Para validar os resultados obtidos foram realizadas comparações entre os resultados encontrados pelo modelo matemático (PIM) apresentado na seção anterior e os obtidos pela heurística apresentada neste trabalho.

Todos os exemplos têm a mesma quantidade de máquinas e tarefas, além do fato de que todas as tarefas devem passar por todas as máquinas uma única vez, sendo esta uma limitação do modelo matemático apresentado.

A heurística desenvolvida neste trabalho contempla mais possibilidade de problemas, como a possibilidade de reentrada de uma tarefa em uma máquina pela qual ela já tenha sido processada e, a não necessidade de todas as tarefas passarem obrigatoriamente por todas as máquinas.

4.1 Problemas 3x3 a 7x7

Para realizar a validação dos resultados, foram gerados e testados 30 exemplos para cada tamanho do problema. Em cada exemplo, na heurística são salvos o melhor resultado obtido (fixado o número de iterações) e o tempo de processamento para obter tal solução, e no modelo matemático, resolvido no Cplex, foi salvo o ótimo obtido pelo mesmo, e o tempo de processamento para obter tal ótimo.

Em cada teste realizado, depois de salvas as informações, foram comparados os resultados do melhor tempo obtido pela heurística em relação ao ótimo obtido pelo método exato, pelas relações:

$$GAP_{solução} = (\text{Melhor resultado heurística})/(\text{Melhor Solução PIM}) - 1$$

Este resultado demonstra o percentual da solução obtida pela heurística em relação à melhor solução obtida pelo modelo matemático. Por exemplo, um GAPsolução = 2,94% indica que o melhor resultado obtido pela heurística para um certo exemplo é 2,94% maior que o valor obtido pelo modelo matemático.

$$GAP_{tempo} = (\text{Tempo heurística})/(\text{Tempo PIM})$$

Sendo que este resultado é o percentual de tempo que a heurística levou para obter a melhor solução em relação ao tempo que o modelo matemático levou para obter a melhor

solução. Por exemplo, um GAPtempo = 4,55% indica que a heurística levou 4,55% do tempo que o modelo matemático para obter a melhor solução.

As médias dos tempos de processamento (em ms) dos exemplos para cada ordem (em ms) dos exemplos para cada ordem estão na tabela a seguir.

Tabela 1 – Tempos médios de processamentos (ms)

Tamanho	Iterações	Heurística	PIM
3x3	100	5,93	63,00
4x4	500	59,90	133,67
5x5	1000	215,53	864,57
6x6	5000	1718,63	32558,67
7x7	5000	2688,57	1628035,67

Fonte: Os autores

Em cada um dos tamanhos processados nesta seção, após todos os testes realizados e salvos todas as informações necessárias, foram tomadas as médias dos GAPs para as soluções e os tempos de processamentos, em cada ordem de problema. Os mesmos estão resumidos na tabela a seguir.

Tabela 2 – Comparativos dos resultados

Tamanho	GAPsolução	GAPtempo
3x3	0,40%	15,26%
4x4	1,57%	52,82%
5x5	1,17%	39,03%
6x6	3,98%	12,15%
7x7	5,29%	1,51%

Fonte: Os autores

4.2 Problemas 8x8 e 9x9

Ao iniciar os testes para os problemas de ordem 8x8, observou-se que o tempo de resolução do modelo matemático para obtenção do ótimo tornou-se inviável, passando de 24 horas o tempo de processamento e ainda assim o ótimo não era atingido. Então, para continuar com a validação dos resultados, foram comparados os melhores valores obtidos pela heurística e a solução que o modelo matemático obteve com 10 vezes mais tempo que o exemplo mais demorado obtido pela heurística.

No caso dos problemas 8x8, o tempo médio de processamento foi de 3606,23ms (todos com 5000 iterações), com desvio padrão de 266,21ms, e o problema mais demorado foi com o tempo de 4193ms. Assim o modelo matemático teve seu tempo de execução limitado em 42s (baseado no maior tempo entre todos os testes).

Os valores negativos para o GAPsolução indicam que a solução encontrada pela heurística com 10\% do tempo para o PIM é menor que a solução obtida pelo modelo

matemático, por exemplo, para o problema 8x8_Ex01, GAPsolução = –4,17% mostra que a solução obtida pela heurística é 4,17% menor que a resposta do modelo matemático. Se o valor for positivo, mostra quanto por cento a resposta da heurística é maior que a resposta do modelo matemático, que foi interrompido no tempo estipulado.

A média dos GAPsolução para os problemas 8x8 é de –3,80%, ou seja, na média, as soluções obtidas pela heurística com 10% do tempo de execução do PIM é 3,80\% melhor.

Pode-se observar o comportamento das soluções na Figura 1, onde a linha em azul com a legenda Resultado Heurística apresenta o melhor valor obtido para o *makespan* pela heurística e a linha em vermelho com a legenda Resultado PIM, para o modelo matemático, com o tempo especificado acima de 42s.

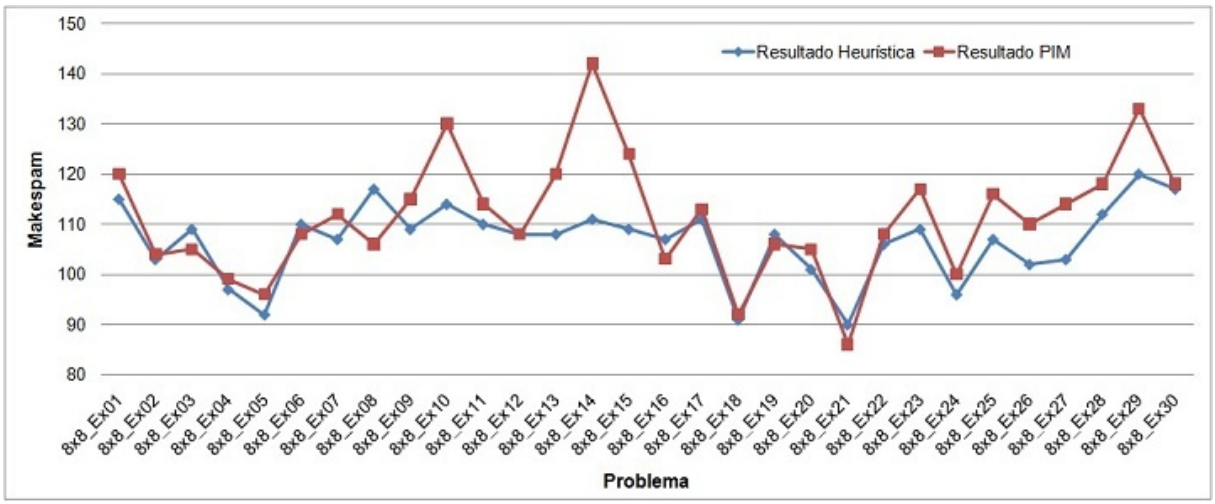

Figura 1 – Resultados 8x8
Fonte: Os autores

Foi efetuada a mesma construção e análise para os problemas 9x9. O tempo médio de execução da heurística nos problemas 9x9 foi de 4744,90ms (também com 5000 iterações), com desvio padrão de 56,08ms, e o problema mais demorado teve o tempo de 4859ms. Assim, o tempo de processamento do modelo matemático foi limitado em 49s (novamente baseado no maior tempo de processamento).

A média dos GAPsolução para os problemas 9x9 é de –24,92%, ou seja, na média, as soluções obtidas pela heurística com 10% do tempo de execução do PIM é 24,92% melhor.

A Figura 2 apresenta os resultados obtidos pela heurística e pelo modelo matemático, valendo os mesmos comentários do caso 8x8. Nota-se que em alguns problemas, como o 9x9_Ex04 não se tem um resultado para o modelo matemático, isto se deve ao fato que o mesmo não encontrou uma primeira solução factível para então começar a otimizar o

problema. Assim, o modelo matemático começou a ficar inviável para tempos pequenos (na ordem de segundos ou poucos minutos) mesmo para obter a primeira solução factível.

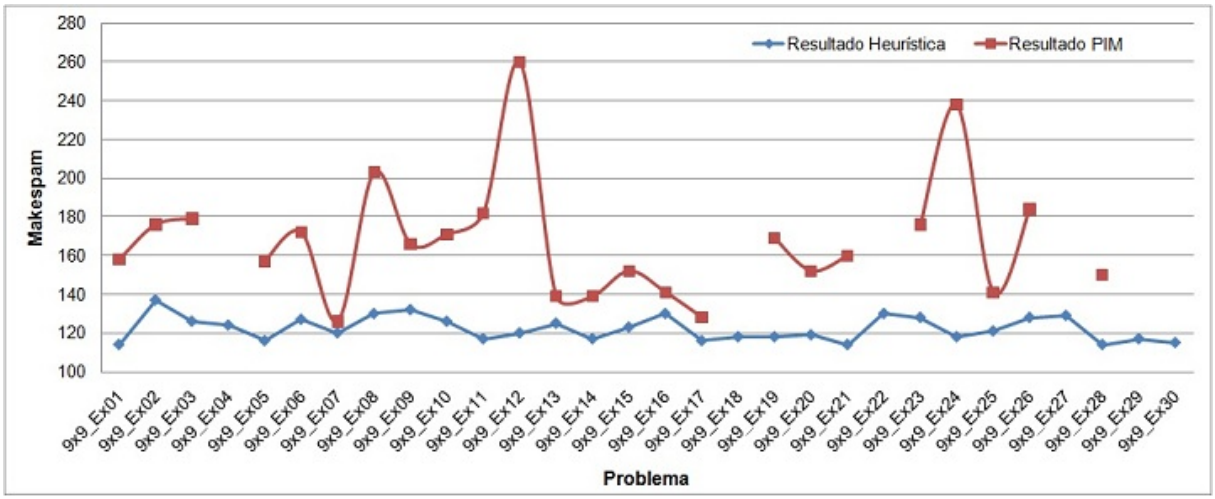

Figura 2 – Resultados 9x9
Fonte: Os autores

Ao final de todos os testes, ainda foi comparado o tempo médio que cada iteração levou para todas as ordens de problema testados. A Tabela apresenta os resultados.

Tabela 3 – Tempos médios das iterações (ms) 3x3 a 9x9

Tamanho	Tempos Médios	Iterações	Tempo/Iteração
3x3	5,93	100	0,06
4x4	59,90	500	0,12
5x5	215,53	1000	0,22
6x6	1718,63	5000	0,34
7x7	2688,57	5000	0,54
8x8	3606,23	5000	0,72
9x9	4744,90	5000	0,95

Fonte: Os autores

Ao final dos testes, foi possível concluir que o objetivo de minimizar o tempo de trabalho foi alcançado pelo modelo. A revisão da literatura foi realizada e nada semelhante foi encontrado para resolver o problema abordado da mesma forma que a heurística utilizada.

O algoritmo foi implementado e, ficou evidenciado que o tempo de processamento é menor, mesmo para os problemas grandes cada iteração dura uma fração de segundo. E não são necessárias muitas iterações para obter uma solução otimizada. Como o processo de cada iteração do algoritmo apresentado é independente da iteração anterior, não há garantias de em qual iteração a melhor solução é obtida. Quanto mais iterações foram dadas ao algoritmo,

apenas há mais possibilidades de respostas para se procurar, em função da aleatoriedade descrita no Passo 4.

Conforme citado na revisão bibliográfica, o problema de sequenciamento contempla muitas possibilidades de problemas. O mesmo procedimento foi aplicado a uma série de testes com estruturas de diversos tamanhos e, optou-se pela escolha da comparação com o resultado exato sobre um modelo de programação linear mista, devido ao fato de não ter sido encontrada uma base de dados com problemas resolvidos de forma heurística. Também, para comparar com a melhor solução possível, a exata, até onde era possível de realizá-la.

7 REFERENCIAS

ALVAREZ-VALDES, R.; FUERTE, A.; TAMARIT, J. M.; GIMENEZ, G.; RAMOS, R. **A heuristic to schedule flexible job-shop in a glass factory**. European Journal of Operational Research, 165, 525-534. 2005.

CHEUNG, W.; ZHOU. H. **Using Genetic Algorithms and Heuristics for Job Shop Scheduling with Sequence-Dependent Setup Times**. Annuals of Operations Research, 107, 65-81, 2001.

COLIN, E. C. **Pesquisa operacional:** 170 aplicações em estratégia, finanças, logística, produção, marketing e vendas. Rio de Janeiro: LTC, 2007. xix, 501 p.

GEORGIADIS, P.; MICHALOUDIS, C. **Real-time production planning and control system for job-shop manufacturing:** A system dynamics analysis. European Journal of Operational Research, 216, 94-104. 2012.

HUANG, K.-L.; LIAO, C.-J. **Ant colony optimization combined with taboo search for the job shop scheduling problem**. Computers and Operational Research, 35, 1030-1046. 2008.

KARIMI-NASAB, M.; SEYEDHOSEINI, S. M. **Multi-level lot sizing and job shop scheduling with compressible process times:** A cutting plane approach. European Journal of Operational Research, 231, 598-616. 2013.

LIU, T.-K.; TSAI, J.-T.; CHOU. J.-H. **Improved genetic algorithm for the job-shop scheduling problems**. International Journal of Advanced Manufacturing Technology, 27, 1021-1029. 2006.

LUO, H.; HUANG, G. Q.; ZHANG, Y.; DAI Q.; CHEN, X. **Two-stage hybrid batching owshop scheduling with blocking and machine availability constraints using genetic algorithm**. Robotic and Computer-Integrated Manifacturing, 25, 962-971. 2009.

MANIKAS, A.; CHANG, Y.-L. **Multi-criteria sequence-dependent job shop scheduling using genetic algorithms**. Computer & Industrial Engineering, 56, 179-185. 2009.

MATI, Y.; DAUZÉRE-PÉRÈS, S.; LAHLOU, C. **A general approach for optimizing regular criteria in the job-shop scheduling problem**. European Journal of Operational Research, 212, 33-42. 2011.

MONKMAN, S. K.; MORRICE, D. J.; BARD, J. F. **A production scheduling heuristic for an electronics manufacturer with sequence-dependent setup costs**. European Journal of Operational Research, 187, 1100-1114. 2008.

NOWICKI, E.; SMUTNICKI, C. **An advanced tabu search algorithm for the job shop problem**. Journal of Scheduling 8, 145-159. 2005.

PARTHANADEE, P.; BUDDHAKULSOMSIRI, J. **Simulation modeling and analysis for production scheduling using real-time dispatching rules:** A case study in canned fruit industry. Computers and Electronics in Agriculture, 70, 245-255. 2010.

PROUD, J. F. **Master scheduling:** practical guide to competitive manufacturing. 2nd ed. New York : J. Wiley & Sons, c1999. 610 p.

QIAN, B.; WANG. L.; HUANG, D.; WANG, X. **Scheduling multi-objective job shops using a memetic algorithm based on di_erential evolution**. International Journal of Advanced Manufacturing Technology, 35, 1014-1027. 2008.

ROCHA, P.L.; RAVETTI, M. G.; MATEUS, G. R. **The metaheuristic GRASP as an upper bound for a branch and bound algorithm in a scheduling problem with nonparallel machines and sequence-dependent setup times**. Proceedings of 4th EU/ME workshop: Design and evaluation of advanced hybrid meta-heuristics, 62-67. Nottingham, UK. 2004.

SADEGHEIH, A. **Scheduling problem using genetic algorithm, simulated annealing and the efects of parameter values on GA performance**. Applied Mathematical Modelling, 30, 147-154. 2006.

SELS, V.; CRAEYMEERSCH, K.; VANHOUCKE, M. **A hybrid single and dual population search procedure for the job shop scheduling problem**. European Journal of Operational Research, 215, 512-523. 2011.

SHEN, L.; BUSCHER, U. **Solving the serial batching problem in job shop manufacturing systems**. European Journal of Operational Research, 221, 14-26. 2012.

TASGETIREN, M. F.; LIANG, Y. C.; SEVKLI, M.; GENCYILMAZ, G. **A particle swarm optimization algorithm for makespam and total flowtime minimization in the permutation flowshop sequencing problem**. European Journal of Operational Research, 177, 1930-1947. 2007.

ZHOU, H.; CHEUNG, W.; LEUNG, L. **Minimizing weighted tardiness of job-shop scheduling using a hybrid genetic algorithm**. European Journal of Research, 194 (1), 637-649. 2009.

Originais recebidos em: 31/01/2016

Aceito para publicação em: 30/06/2016