ISSN -

BUSCA TABU PARA A MINIMIZAÇÃO DO TEMPO TOTAL DE ATRASO NO PROBLEMA DE FLOWSHOP

Débora Prettí Ronconi
USP
Vinícius Amaral Armentano
UNICAMP

Resumo

Este trabalho propõe uma heurística para o problema de programação de tarefas no ambiente *flowshop* com o objetivo de minimizar a soma dos atrasos. O método utiliza Busca Tabu como estratégia de exploração do espaço de soluções. Duas vizinhanças são utilizadas: Troca e Inserção. Avalia-se o efeito de diferentes soluções iniciais, dentre estas, adaptações de regras de despacho e do algoritmo NEH. Testes computacionais são apresentados, e realiza-se uma comparação com o algoritmo NEH.

Palavras-chave: programação de tarefas; *flowshop;* regras de despacho; busca tabu, tempo de atraso.

Abstract

This work proposes a heuristic for the flowshop scheduling problem with the objective of minimizing total tardiness. The method uses Tabu Search to explore the space of solutions. Two strategies of neighborhood generation are used: Swap and Insertion. The effect of initial solutions given by dispatching rules and NEH algorithm is evaluated. Computational tests are reported and a comparison is made with NEH algorithm.

Keywords: scheduling; flowshop; dispatching rules; tabu search; tardiness.

1. Introdução

O objetivo deste trabalho é propor um método baseado em Busca Tabu para a resolução do problema de programação de tarefas *(scheduling)* no ambiente *flowshop* com o objetivo de minimizar a soma dos atrasos em relação às datas de entrega. Este problema consiste de **n** tarefas que devem ser processadas por **m** máquinas ordenadas. Em outras palavras, a j-ésima operação de todas as tarefas deve ser sempre feita na máquina j. O tempo de processamento das operações e as datas de entrega das tarefas são conhecidos.

Existem $(n!)^m$ diferentes alternativas para ordenar **n** tarefas em **m** máquinas. Porém, na maioria das pesquisas considera-se somente um subconjunto destas alternativas, partindo do principio que a sequência das tarefas nas máquinas é a mesma. Neste caso, o número de alternativas é reduzido para (n!), e os programas correspondentes são chamados de programas de permutação. O nível de dificuldade deste problema pode ser estimado através de um caso particular cuja resolução é comprovadamente complexa: o problema com uma máquina é NP-*hard* para o critério de minimização da soma dos atrasos das tarefas (Du&Leung [3]).

A literatura envolvendo o critério de minimização da soma dos atrasos não é muito vasta. Boas revisões bibliográficas podem ser encontradas em Koulamas[11] e Kim [9]. Gelders & Sambadam[4], e Ow[14] apresentam heurísticas baseadas em cálculos de prioridades entre tarefas com o objetivo de minimizar a soma ponderada dos atrasos. Ow desenvolve um procedimento para *flowshop* proporcional, um ambiente no qual os tempos de processamento de cada tarefa nas máquinas são proporcionais. Uma regra de prioridades, denominada regra do tempo ocioso, é aplicada à máquina mais lenta (gargalo).

Kim[9] avalia diversas heurísticas para a resolução deste problema, dentre estas uma adaptação do algoritmo de Nawaz et al. [13], conhecido como algoritmo NEH. Inicialmente este algoritmo necessita de uma ordenação das tarefas de acordo com algum critério preestabelecido, de forma a criar uma lista de prioridades entre estas. A seguir, realiza-se uma enumeração parcial das possíveis soluções baseada nesta ordenação. Kim recomenda a utilização do algoritmo NEH, com a regra EDD (ordem não-decrescente de datas de entrega) para a construção da lista inicial, dado que este forneceu bons resultados num tempo viável.

Buscando encontrar soluções ótimas Kim[8] e Sen et al. [17] propõem algoritmos baseados na técnica *Branch-and-Bound* para ao problema *de flowshop* com duas máquinas. Sen et al. desenvolvem um teorema baseado na troca de pares adjacentes de tarefas, e a partir deste obtém uma forma analítica de avaliar a validade de uma possível troca. Kim apresenta um novo limitante inferior para a soma dos atrasos, além de uma nova regra de dominância. Testes computacionais em problemas gerados aleatoriamente mostraram que este algoritmo supera o algoritmo de Sen et al. Kim[10] utiliza estas e outras propriedades para a resolução de problemas *de flowshop* com **m** máquinas. Townsend[20] também desenvolveu um *Branch-and-Bound*, porém, com o objetivo de obter a solução ótima para o critério do atraso máximo.

A utilização da metaheurística Busca Tabu em problemas de *flowshop é* uma idéia razoavelmente nova. A primeira proposta com este enfoque foi apresentada por Widmer & Hertz[21]. Estes autores desenvolveram uma heurística com o objetivo de minimizar o *makespan.* Um bom trabalho, também para minimizar o *makespan,* foi realizado por Taillard [18], destacando-se o extensivo estudo para a

definição dos elementos da busca. O autor compara o resultado do algoritmo proposto com o obtido através de uma implementação computacional eficiente do algoritmo NEH, e conclui que a Busca Tabu obtém os melhores resultados. Reeves[16] propõe uma exploração parcial da vizinhança com o objetivo de ampliar o número de soluções obtidas. O desempenho deste algoritmo superou o proposto por Tailiard.

As aplicações de Busca Tabu com o objetivo específico de minimizar a soma dos atrasos são poucas. Adenso_Diaz [1] trata o problema do *flowshop* proporcional, e utiliza como solução inicial a heurística proposta por Ow [14]. A contribuição deste autor se constitui na redução de duas vizinhanças geradas por movimentos normalmente utilizados: o de inserção e o de troca de tarefas. No estudo comparativo apresentado por Kim [9], a implementação de Busca Tabu proposta por Widmer & Hertz [21] apresenta em média os melhores resultados, porém com um tempo computacional alto.

Neste trabalho apresenta-se uma heurística baseada em Busca Tabu para o critério de minimização da soma dos atrasos das tarefas. A seção a seguir contém uma descrição da heurística proposta e na seção 3 apresentam-se os resultados.

2. Heurística Proposta

A heurística proposta é composta por uma busca em vizinhança orientada por Busca Tabu. Esta técnica é descrita por Glover[5][6] e objetiva guiar a busca na exploração do espaço de soluções de um problema, além da otimalidade local. Um dos principais recursos desta técnica é a proibição da ocorrência de certos movimentos por um número determinado de iterações. Este recurso tem o intuito de evitar ciclagens e atingir pontos diversificados no espaço de soluções.

2.1 Solução Inicial

Avaliou-se para solução inicial adaptações de regras de despacho e do algoritmo NEH, considerado o método mais eficiente para o critério do atraso médio (Kim[9]). Dado que o algoritmo NEH necessita de uma regra que ordene inicialmente as tarefas, optou-se pela utilização das mesmas regras de despachos que são avaliadas como solução inicial para Busca Tabu. Estas regras foram selecionadas por serem as mais simples ou conhecidas por trabalharem melhor que outras (Kim[9]). Cada regra se baseia numa medida diferente que é calculada, se necessário, a cada instante de tempo t. Ordena-se as tarefas em ordem não-decrescente destas medidas. As seguintes regras são selecionadas:

-EDD: d_i
-SLACK: $d_i - C_i(\sigma)$
-MDD: $\max(d_i, C_i(\sigma))$

onde,

$\mathbf{d_i}$ é a data de entrega da tarefa **i**;
σ é a seqüência de tarefas já estabelecida no instante **t**;
$C_i(\sigma)$ é o tempo de finalização da tarefa **i**, $\mathbf{i} \notin \sigma$, se esta for seqüenciada logo após a seqüência σ.

Além destas sugere-se a utilização de um limitante inferior do atraso de cada tarefa (LI) como regra para ordenação das tarefas. Este limitante agrega mais informações que a regra EDD, mantendo a simplicidade do cálculo.

-LI: $d_i - \sum_j P_{ij}$

onde,

$\mathbf{p_{ij}}$ é o tempo de processamento da tarefa **i** na máquina **j**.

2.2 Busca Tabu

Busca Tabu é uma estratégia que requer a definição de seus componentes básicos. Dentre estes destacam-se: estratégia de geração de vizinhanças, escolha da próxima solução vizinha, seleção dos atributos do movimento, regras de proibição, critério de parada, critério de liberação tabu (critério de aspiração), e número de iterações que um movimento será tabu (tamanho da lista).

Com o intuito de determinar estes componentes, foram realizadas diferentes atribuições para cada um deles. Estas são descritas a seguir:

Estratégia de geração de vizinhanças:

* inserção de tarefas: uma tarefa **i**, localizada na posição π**(i)**, é inserida na posição **y**; esta estratégia gera uma vizinhança de tamanho $\mathbf{(n-1)^2}$;
* troca de tarefas: uma tarefa **i**, localizada na posição π**(i)**, é trocada com a tarefa **j**, localizada na posição π**(j)**; neste caso o tamanho da vizinhança é $\frac{\mathbf{N}}{\mathbf{2}}\mathbf{(n-1)}$.

Escolha da próxima solução, dada uma solução avalia-se cada solução vizinha e seleciona-se:

* a primeira solução: que melhora a solução atual (Primeiro Movimento); ou
* a melhor solução encontrada (Melhor Movimento).

Seleção dos atributos do movimento e regras de proibição: para cada tipo de movimento selecionou-se atributos e regras especificas. Estes são apresentados a seguir, do mais restrito ao menos restrito. Para o movimento de inserção tem-se:

- **i** - a tarefa **i** não pode ser movida;
- **i** - a tarefa **i** não pode ser escolhida para inserção;
- **i,** π**(i)** - a tarefa **i** não pode retomar à posição π**(i)**.

Para o movimento de troca, estabeleceu-se como atributo e respectiva regra de proibição:

- **ij** - a tarefa **i** e a tarefa **j** não podem ser movidas.
- **i** - a tarefa **i** não pode ser movida;
- **mi** - a tarefa **i** não pode ser movida para posições **k,** onde $\mathbf{k} \leq \pi\mathbf{(i)}$;

Estes atributos e regras foram selecionados por serem os mais conhecidos ou por apresentarem melhores resultados em problemas correlacionados como em Laguna et al. [12] e Tailard [18].

Critério de parada: tempo de C.P.U.. Estabeleceu-se este critério com o intuito de realizar comparações com diferentes combinações de componentes que utilizam, seguramente, um número diferenciado de iterações. Apresenta-se na Tabela 1 abaixo o tempo estabelecido para cada dimensão de problema, onde **n** e **m** são respectivamente o número de tarefas e o número de máquinas.

Tabela 1: Tempos utilizados para cada dimensão.

Dimensão (nxm)	Tempo (seg.)
20 x 05	25
20 x 10	50
20 x 20	90
50 x 05	200
50 x 10	500
50 x 20	1000

Estes tempos foram estabelecidos através de testes, onde observou-se que estes eram suficientes para um bom desempenho da heurística.

Critério de aspiração: um movimento pode ser liberado da condição tabu caso possua um valor de solução melhor que a melhor solução encontrada até o momento pela busca.

Tamanho da lista: a atribuição da duração da aplicação de uma regra de proibição a um movimento é feita de forma dinâmica. Estabeleceu-se para cada combinação atributo-regra uma faixa de geração uniforme desta duração, com as seguintes formas de atualização:

- **Aleatória -** a cada iteração gera-se um novo valor para a duração da proibição.
- **Ajustada -** a cada vinte iterações gera-se um novo valor para a duração da proibição, e a cada iteração é realizado um ajuste neste valor . Este ajuste é feito de acordo com o valor da solução selecionada com relação à solução que gerou o movimento (solução atual). Caso o valor da solução selecionada seja maior que o da solução atual, a duração da proibição associada à solução selecionada é aumentada de uma unidade. Caso o valor da solução selecionada seja menor que o da solução atua], a duração é diminuída de uma unidade. Um procedimento similar é proposto em Dell'Amico & Trubian[2].

A Tabela 2 mostra os limitantes da faixa de geração uniforme de acordo com a combinação utilizada.

Tabela 2: Limitantes de geração da lista tabu.

Movimento	Atributo	Regra	Limitante Inferior	Limitante Superior
Inserção	**i**	**i** não pode ser movido	N/4	• N/2
	i	**i** não pode ser escolhido para inserção	N/2	*N
	i, π (**i**)	**i** não pode retornar a posição π (**i**)	4N	• 5N
Troca	**ij**	**i** e **j** não podem ser movidos	N/4	• N/2
	i	**i** não pode ser movido	N/2	*N-1
	mi	**i** não pode ser movido para posições **k**, onde **k** $\leq\pi$(**i**)	N/2	• 3N/2

Observa-se que quanto mais restrita é a combinação atributo-regra menores são os limitantes superiores. Os limitantes marcados com asterisco foram selecionados através de um estudo sobre o número de iterações necessárias para que não houvessem mais movimentos disponíveis (não tabu) para serem escolhidos, considerando que se uma regra de proibição é ativada, esta permanece válida ao longo das iterações. Além disso, assume-se que o critério de aspiração não está ativo. Este estudo, obviamente, está diretamente relacionado com o tipo de atributo e regra ativos. Para exemplificar, apresenta-se a seguir a determinação do limitante de geração para o atributo **i** do movimento de troca. Suponha que uma dada tarefa foi escolhida para ser trocada. Como a regra de proibição estabelece que a tarefa selecionada não deve mais ser movimentada, conclui-se que tem-se agora **n-1** tarefas a serem analisadas para troca. Considerando que se proíbe uma tarefa a cada iteração, e que para realização do movimento de troca há necessidade de existirem pelo menos duas tarefas liberadas, a última troca possível se realizará na iteração **n-1**.

Os limitantes superiores marcados com pontos e os limitantes inferiores foram estimados através de testes computacionais. Na próxima seção apresentam-se os resultados computacionais dos testes realizados com as soluções iniciais e com a aplicação de Busca Tabu.

3. Resultados Computacionais

3.1 Características da Implementação

Os programas computacionais foram escritos na linguagem C e os testes realizados numa estação de trabalho SUN modelo SPARCstation classic. Os dados foram gerados conforme sugerido em

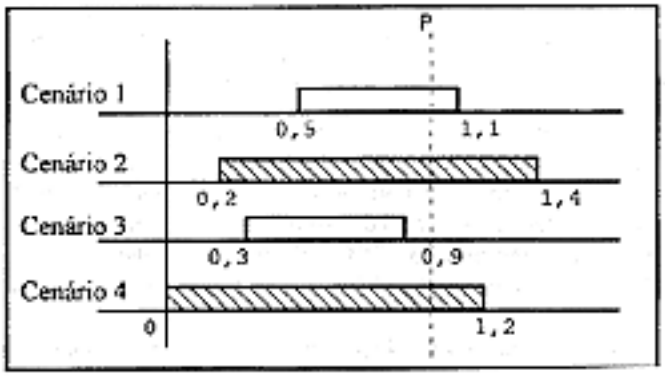
Taillard [18], de tal forma a torná-los independentes do computador utilizado. Gerou-se três diferentes problemas para cada uma das seis dimensões escolhidas (Tabela 1). Para cada problema foram selecionados quatro distintos cenários, ou seja, foram testados 6 x 3 x 4 = 72 problemas. O tempo de processamento é uniformemente distribuído entre 0 e 100, e a data de entrega também é uniformemente distribuída entre P(1- T -R/2) e P(1 - T + R/2) (Potts & Wassenhove [15]), onde P é um limitante inferior do *makespan* (Taillard[18]), definido por:

$$P = \max\left\{\max_{1\le j\le m}\left\{\sum_{i=1}^{n} p_{ij} + \min_{i}\sum_{l=1}^{i-j} p_{il} + \min_{i}\sum_{l=j+1}^{m} p_{il}\right\}, \max_{i}\sum_{j=1}^{m} p_{ij}\right\}$$

e, T e R são respectivamente o fator de atraso das tarefas, e a faixa de dispersão da data de entrega Os seguintes cenários foram selecionados (Figura l):

cenário l: baixo fator de atraso (T=0,2) e pequena faixa de datas de entrega (R=0,6)
cenário 2: baixo fator de atraso (T=0,2) e larga faixa de datas de entrega (R=l ,2)
cenário 3: alto fator de atraso (T=0,4) e pequena faixa de datas de entrega (R=0,6)
cenário 4: alto fator de atraso (T=0,4) e larga faixa de datas de entrega (R=l,2)

Figura l: Representação dos cenários gerados



Devido a dificuldade de obter soluções ótimas para estes problemas, utilizou-se como medida de desempenho da heurística o índice de desvio relativo (Kim[9]). Este índice (IDR) é calculado da seguinte forma:

$$IDR = \frac{T_a - T_{menor}}{T_{maior} - T_{menor}}$$

onde T_a é a solução do método que está sendo avaliado, e T_{menor} e T_{maior} são respectivamente a melhor e a pior solução encontradas. A faixa de variação deste índice é de 0 a l. Este índice é extremamente útil para a avaliação entre diferentes métodos pois posiciona cada método entre a menor e a maior solução dentre todas avaliadas.

3.2 Avaliação das Soluções Iniciais

Primeiramente implementou-se as regras de despacho e o algoritmo NEH que são utilizados como solução inicial para a heurística. Nesta etapa de testes o desempenho dos algoritmos foi analisado em problemas maiores conforme apresentados na Tabela 3. Para cada dimensão gerou-se dez diferentes problemas. Além disso variou-se o cenário como descrito na seção 3.1. A Tabela 3 mostra o resultado desta primeira fase das implementações.

Tabela 3: Comparação entre regras de despacho e o algoritmo NEH

N	M	EDD	LI	MDD	SLACK	EDD+ NEH	LI+ NEH	MDD+ NEH	SLACK +NEH
20	5	0,584	0,897	0,383	0,748	0,069	0,044*	0,077	0,077
20	10	0,712	0,937	0,432	0,818	0,051	0,022*	0,080	0,043
20	20	0,805	0,964	0,507	0,896	0,036*	0,037	0,064	0,049
50	5	0,606	0,787	0,402	0,660	0,057	0,054*	0,089	0,064
50	10	0,755	0,862	0,607	0,825	0,062	0,046*	0,082	0,069
50	20	0,861	0,932	0,662	0,894	0,064	0,027*	0,106	0,054
100	5	0,529	0,654	0,391	0,598	0,050	0,025*	0,046	0,042
100	10	0,725	0,788	0,578	0,732	0,053	0,024*	0,070	0,051
100	20	0,797	0,921	0,631	0,859	0,070	0,036*	0,071	0,063
200	10	0,603	0,671	0,490	0,600	0,050	0,033*	0,048	0,037
MÉDIA		0,698	0,841	0,508.	0,763	0,056	0,035*	0,073	0,055
DESVIO PADRÃO		0,111	0,112	0,100	0,117	0,001	0,001	0,003	0,002

Os asteriscos na Tabela 3 indicam as melhores soluções. Observa-se através destes e da média dos índices que o algoritmo NEH associado à regra LI obteve os melhores resultados. A associação deste com as outras regras, embora mostre um desempenho um pouco inferior, também apresentou bons resultados. Destaca-se o fato do tempo computacional das regras ser pequeno; estas solucionam os problemas de maior dimensão (200x10) num tempo médio de 1,7 segundos enquanto o algoritmo NEH soluciona estes mesmos problemas num tempo médio de 118 segundos.

O intuito desta análise é encontrar soluções iniciais com um bom equilíbrio entre qualidade e tempo computacional. Logo, como com regras de despacho tem-se soluções de pior qualidade e tempos excelentes, e com o algoritmo NEH tem-se soluções de melhor qualidade e tempos maiores, conclui-se que não se pode, nesta etapa do trabalho, eliminar nenhuma das alternativas analisadas.

Implementou-se a seguir uma versão de Busca Tabu com objetivo de avaliar o efeito das diferentes soluções iniciais sobre a solução final obtida pelo método. Os componentes desta versão

são descritos a seguir. Dada uma tarefa i (i=1,..,n) avalia-se todas as possíveis trocas. Seleciona-se como a próxima solução o primeiro movimento de troca que melhora a solução atual. Proíbe-se então, que a tarefa i escolhida se mova por um período de 7 iterações. Admite-se um movimento proibido (tabu) caso este melhore a função objetivo. Interrompe-se a execução da busca depois de N (número de tarefas) iterações sem melhoria na função objetivo ou 250 iterações. Apresenta-se na Tabela 4 abaixo o desempenho desta implementação.

Tabela 4: Busca Tabu e diferentes soluções iniciais

N	M	EDD	LI	MDD	SLACK	EDD+ NEH	LI+ NEH	MDD+ NEH	SLACK+ NEH
20	5	0,376	0,284	0,309	0,499	0,256	0,234	0,229	0,166*
20	10	0,465	0,336	0,474	0,427	0,333*	0,414	0,421	0,355
20	20	0,551	0,484	0,426	0,457	0,374	0,356	0,399	0,317*
50	5	0,226	0,274	0,280	0,239	0,263	0,242	0,187*	0,234
50	10	0,560	0,487	0,584	0,449	0,414	0,330	0,322*	0,458
50	20	0,550	0,576	0,495	0,622	0,377	0,313	0,304	0,295*
MÉDIA		0,455	0,407	0,428	0,449	0,336	0,315	0,310	0,304*
DESVIO PADRÃO		0,121	0,115	0,106	0,113	0,059	0,063	0,084	0,092

Analisando a Tabela 4, nota-se que a regra SLACK associada ao algoritmo NEH apresentou em média os melhores resultados como solução inicial para heurística. Porém, destaca-se o fato das médias possuírem valores próximos, além do desvio padrão associado a estas ter um valor relativamente alto. Conseqüentemente pode-se concluir que as soluções iniciais avaliadas tiveram um desempenho equivalente. Logo, considerando o alto tempo computacional do algoritmo NEH, quando comparado às regras de despacho, optou-se por sua eliminação como gerador da solução inicial na próxima fase dos testes.

3.3 Aplicação de Busca Tabu

Na seção 2.2 foram definidos os componentes testados na Busca Tabu. Com o intuito de avaliar todas as possíveis combinações, gerou-se 96 diferentes implementações variando-se: solução inicial, estratégia de geração de vizinhanças, escolha da próxima solução, seleção dos atributos do movimento, regras de proibição, e tamanho da lista.

A primeira avaliação realizada considerou a solução inicial. As regras de despacho como solução inicial apresentaram, de forma geral, um desempenho equivalente, tanto no movimento de troca quanto no movimento de inserção.

Quanto aos atributos e regras de proibição associadas a estes, destacou-se dentre todas as combinações realizadas, o atributo **i**, com proibição da tarefa **i** ser selecionada para inserção, e o atributo **mi** como os melhores atributos, respectivamente, nos movimentos de inserção e troca. Observe que para a vizinhança gerada pela troca o atributo que obteve melhor desempenho é o atributo menos restrito, o que é coerente dado que a vizinhança da troca não é grande comparativamente à da inserção. Por outro lado, o melhor atributo para a vizinhança gerada pela inserção é um atributo razoavelmente restritivo, o que também é coerente dado o tamanho da vizinhança.

Em relação à estratégia de geração de vizinhanças, observou-se que o movimento de inserção obteve resultados superiores ao movimento de troca em todas as combinações. Este fato ocorre devido à inserção possuir uma vizinhança mais rica em termos de alteração de características da solução atual, o que teve muita influência no seu bom desempenho. Enquanto a troca altera o programa somente nas posições selecionadas para o movimento, a inserção desloca todas as tarefas que se localizem entre a tarefa escolhida para inserção e sua nova localização. A Tabela 5 apresenta a porcentagem de melhoria, em relação à associação LI+NEH, da melhor combinação de componentes que utiliza o movimento de troca. Esta porcentagem de melhoria é calculada da seguinte forma:

$$Porcentagem\ de\ melhoria = \frac{T_{li+neh} - T_{tabu}}{T_{li+neh}}$$

A combinação utilizando o movimento de troca consiste dos seguintes componentes: regra EDD como solução inicial; Melhor Movimento; atributo **mi**; e lista dinamicamente ajustada.

Tabela 5: Melhoria Percentual da Busca Tabu com troca em relação ao NEH

Dimensão (n x m)	Cenários			
	1	2	3	4
20x5	80,26	22,24	28,01	16,35
20x10	15,55	8,05	5,56	9,90
20x20	2,69	5,28	3,86	2,22
50x5	96,13	100,00	30,27	27,88
50x10	72,49	12,92	12,14	6,65
50x20	36,80	5,47	3,03	4,14
MÉDIA	50,65	25,66	13,81	11,19
DESVIO PADRÃO	34,51	33,75	11,24	8,73

Os melhores resultados de inserção possuem os seguinte componentes: solução inicial fornecida pela regra MDD; geração de vizinhança através do movimento de inserção. Melhor Movimento; atributo **i** com proibição da tarefa i ser selecionada para inserção; e lista dinamicamente ajustada.

Esta implementação, quando comparada com a associação LI+NEH, obteve as porcentagens de melhoria apresentadas na Tabela 6 a seguir.

Tabela 6: Melhoria percentual da Busca Tabu com inserção em relação ao NEH

Dimensão (n x m)	Cenários			
	I	2	3	4
20x5	83,94	25,52	31,78	19,00
20x10	27,83	9,44	9,99	12,36
20x20	9,23	7,91	6,80	4,76
50x5	96,13	100,00	32,29	34,77
50x10	84,58	21,33	22,65	23,27
50x20	50,15	13,70	8,38	9,72
MÉDIA	58,64	29,65	18,98	17,31
DESVIO PADRÃO	32,10	32,07	11,20	9,86

A heurística utilizando a vizinhança de troca obteve em média uma melhoria de 25,33%, enquanto a utilização da inserção gerou soluções com 31,15% de melhoria, o que confirma o melhor desempenho da vizinhança de inserção.

Através da Tabela 5 e 6 nota-se que nos cenários l e 2, que possuem baixo fator de atraso, a heurística obteve grandes melhorias em relação ao algoritmo NEH, enquanto nos cenários 3 e 4, que possuem alto fator de atraso, a melhoria é pequena. Não se pode, no entanto, avaliar o desempenho de Busca Tabu com respeito ao fator de atraso baseado nos resultados das Tabelas 5 e 6, pois uma maior melhoria em relação ao algoritmo NEH não implica que" as soluções fornecidas por Busca Tabu estejam próximas das soluções ótimas.

Com o intuito de avaliar o desempenho da heurística, foi implementado o *Branch-and-Bound* proposto por Kim[10]. A precisão utilizada foi de 5% e a solução inicial dada foi a obtida através da Busca Tabu. Após 12 horas de execução para cada problema, verificou-se que o resultado fornecido pelo *Branch-and-Bound* não superou nenhuma solução heurística. Dentre os 72 problemas testados, encontrou-se 10 soluções ótimas com precisão de 0,5%.

Para comprovar o comportamento do método, a melhor combinação utilizando inserção foi novamente avaliada sobre um novo conjunto de 120 problemas.

Tabela 7: Melhoria percentual da Busca Tabu em relação ao NEH
120 novos problemas

Dimensão (n x m)	Cenários			
	1	2	3	4
20x5	75,70	3,98	17,80	19,77
20x10	37,58	9,09	10,68	10,32
20x20	9,04	10,26	4,37	5,26
50x5	98,32	79,03	24,37	16,53
50x10	87,80	37,60	24,77	16,30
50x20	44,38	17,60	10,09	9,78
MÉDIA	58,80	26,26	15,35	12,99
DESVIO PADRÃO	31,15	25,93	7,60	4,94

Observa-se da Tabela 7, que em média a melhoria porcentual de Busca Tabu em relação ao algoritmo NEH, para cada cenário, se mantém. Logo conclui-se que a heurística tem um desempenho estável.

4. Conclusões

Neste trabalho apresentou-se uma heurística para o problema de *flowshop* com o objetivo de minimizar a soma dos tempos de atraso. Utilizou-se como estratégia para guiar esta heurística a técnica Busca Tabu. Obteve-se bons resultados com a implementação básica desta técnica. Ressalta-se que em todos os problemas a heurística superou as soluções apresentadas pelo algoritmo NEH, ao contrario de Kim[9] onde a Busca Tabu teve um desempenho inferior a outras heurísticas, incluindo o algoritmo NEH, em 25% dos problemas testados. Espera-se melhorias através de implementações mais avançadas como intensificação e diversificação (Glover et al. [7]).

Referências bibliográficas

1. Adenso Díaz, B., 1992, *Restricted neighborhood in the tabu search for the flow shop problem,* European Journal of Operational Research, 62, pp. 27-37.

2. Dell'Amico, M., & Trubian, M., 1993, *Applying tabu search to the job-shop scheduling problem,* Annals of Operations Research, 41, pp. 231-252.

3. Du, J., & Leung, J.Y.-T., 1990, *Minimizing total tardiness on one machine is NP-hard,* Mathematics of Operations Research, 15, pp. 483-495.

4. Gelders, L.F., & Sambandam, N., 1978, *Four simple heuristics for scheduling a flow-shop,* International Journal of Production Research., 16, pp. 221-231.

5. Glover, F.W., 1989, *Tabu Search, Part I,* ORSA Journal on Computing, 13, pp. 190-206.

6. Glover, F.W., 1990, *Tabu Search, Part II,* ORSA Journal on Computing, 2.1, pp. 4-32.

7. Glover, F.W. & Laguna, M., 1993, *Tabu Search,* Modern Heuristic Techniques for Combinatorial Problems, C. Reeves, ed., Blackwell Scientific Publishing, pp.70-150.

8. Kim, Y-D., 1993a, *A New Branch and Bound Algorithm for Minimizing Mean Tardiness in Two-Machine Flowshops,* Computers and Operations Research., 20, pp. 391-401.

9. Kim, Y-D., 1993b, *Heuristics for Flowshop Scheduling Problems Minimizing Mean Tardiness,* Journal of the Operational Research Society, 44.1, pp. 19-28.

10. Kim, Y-D., 1995, *Minimizing mean tardiness on flowshops,* European Journal of Operational Research, 85.3, pp. 541-555.

11. Koulamas, C., 1994, *The Total Tardiness Problem: Review and Extensions,* Operations Research, 42.6, pp. 1025-1041.

12. Laguna, M., Barnes, J.W., & Glover, F.W., 1991, *Tabu search methods for a single machine scheduling problem,* Journal of Intelligent Manufacturing, 2, pp. 63-74.

13. Nawaz, M., Enscore, E.E., & Ham I., 1983, *A Heuristic Algorithm for the m-Machine, n-Job Flow-shop Sequencing Problem,* OMEGA, 11.1, pp. 91-95.

14. Ow, P.S., 1985, *Focused scheduling in proportional flowshops,* Management Science, 31, pp. 852-869.

15. Potts, C.N., & Van Wassenhove, L.N., 1982, *A Decomposition Algorithm for the Single Machine Total Tardiness Problem,* Operations Research Letters, 1.5, pp. 177-181.

16. Reeves, C.R.**,** 1993, *Improving the Efficiency of Tabu Search for Machine Sequencing Problems,* Journal of the Operational Research Society, 44.4, pp.375-382.

17. Sen, T., Dileepan, P, & Gupta, J.N.D., 1989, *The two-machine flowshop scheduling problem with total tardiness,* Computers and Operations Research., 16, pp. 333-340.

18. Tailiard, E.,1990, *Some efficient heuristic methods for the flow shop Sequencing,* European Journal of Operational Research, 47, pp. 65-74.

19. Tailiard, E.,1993, *Benchmarks for basic scheduling problems,* European Journal of Operational Research, 64, pp. 278-285.

20. Townsend, W., 1977, *Sequencing n jobs on m machines to minimize maximum tardiness: a branch-and-bound solution,* Management Science, 23, pp. 1016-1019.

21. Widmer, M., & Hertz, A., 1989, *A new heuristic method for the flow shop sequencing problem,* European Journal of Operational Research, 41, pp. 186-193.

Republicado de Pesquisa Operacional, v.15, n. 1 e 2, pp. 23-24, 1995