Revista Produção On Line
Universidade Federal de Santa Catarina
Florianópolis – SC - Brasil
www.producaoonline.inf.br
ISSN 1676 - 1901 / Vol. X/ Num. X/ mês de ano

AVALIAÇÃO DE MÉTODOS HEURÍSTICOS CONSTRUTIVOS PARA O PROBLEMA DE PROGRAMAÇÃO DE OPERAÇÕES *NO-WAIT FLOW SHOP*

EVALUATION OF CONSTRUCTIVE HEURISTICS FOR NO-WAIT FLOW SHOP SCHEDULING

Fábio José Ceron Branco

Mestre em Engenharia de Produção

Universidade de São Paulo - Escola de Engenharia de São Carlos

Departamento de Engenharia de Produção

Avenida Trabalhador Sãocarlense, 400 - Centro, São Carlos - SP - Brasil - 13566-590

Telefone: (16) 3373-9428, E-mail: fbranco@hotmail.com

Marcelo Seido Nagano

Professor Doutor

Universidade de São Paulo - Escola de Engenharia de São Carlos

Departamento de Engenharia de Produção

Avenida Trabalhador Sãocarlense, 400 - Centro, São Carlos - SP - Brasil - 13566-590

Telefone: (16) 3373-9428, E-mail: drnagano@usp.br

João Vitor Moccellin

Professor Titular

Universidade de São Paulo - Escola de Engenharia de São Carlos

Departamento de Engenharia de Produção

Avenida Trabalhador Sãocarlense, 400 - Centro, São Carlos - SP - Brasil - 13566-590

Telefone: (16) 3373-9428, E-mail: jvmoccel@sc.usp.br

Revista Produção On Line
Universidade Federal de Santa Catarina
Florianópolis – SC - Brasil
www.producaoonline.inf.br
ISSN 1676 - 1901 / Vol. X/ Num. X/ mês de ano

RESUMO

Este artigo trata do problema de programação de operações em um ambiente de produção *no-wait flow shop*, tendo como objetivo a minimização da duração total da programação. A partir da análise das características do problema um novo método heurístico construtivo é apresentado pela combinação de diferentes procedimentos de ordenações iniciais e de re-seqüenciamentos das tarefas encontradas na literatura. Os resultados experimentais mostram a superioridade do novo método para o conjunto de problemas tratados em relação a qualidade da solução obtida e esforço computacional.

Palavras chave: Programação da produção, *No-Wait Flow Shop*, Métodos heurísticos.

ABSTRACT

In this article we consider the no-wait flow shop problem with makespan objective. Following an investigation of the problem characteristics a new constructive heuristic method is presented by the combination with other improvement strategies reported in the literature. Experimental results shows that the new heuristic provide better solutions regarding both the solution quality and computational effort.

Keywords: *Production scheduling, No-Wait Flow Shop, Heuristics.*

1. INTRODUÇÃO

O problema tradicional de Programação *Flow Shop* é um problema de produção onde um conjunto de *n* tarefas deve ser processado, na mesma seqüência, por um conjunto de *m* máquinas. Quando a ordem de processamento em todas as máquinas for a mesma, tem-se o ambiente de produção *Flow Shop* Permutacional, onde o número de possíveis programações para *n* tarefas é *n*!.

Uma situação específica e muito freqüente para este problema é a programação da produção em sistema de produção *flow shop* sem interrupção na execução das tarefas, que ocorre geralmente em um ambiente caracterizado pela tecnologia do processo. Algumas aplicações típicas para este problema são encontradas em indústrias de produção de metais e plásticos (WISMER, 1972), indústrias químicas (RAJENDRAN, 1994), farmacêuticas (RAAYMAKERS e HOOGEVEEN, 2000) e indústrias de processamento de alimentos (HALL e SRISKANDARAYAH, 1996).

O problema consiste em obter uma seqüência das tarefas que otimize uma determinada medida de desempenho. Nos modelos para solução do problema, as medidas usuais referem-

se à minimização do tempo médio de fluxo (*mean flow time*), associado à redução do estoque em processamento, e à minimização da duração total da programação (*makespan*) associada à utilização eficiente dos recursos produtivos, sendo esta última a medida adotada neste trabalho.

Conforme a literatura este problema também é chamado de "*Flow Shop* sem espera" ou "*No-Wait Flow Shop*" (*NWFS*) apresentando a seguinte estrutura conforme apresentada na figura 1.

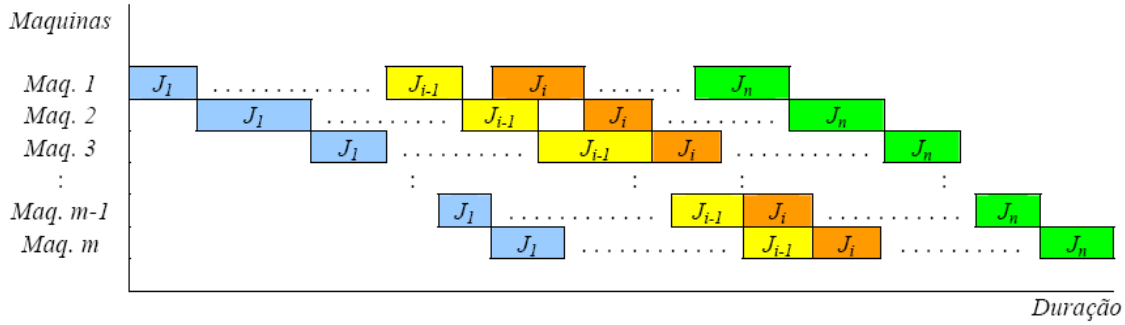

Figura 1 - *No-Wait Flow Shop* com *m* máquinas e *n* tarefas

A figura 1 apresenta a programação de um conjunto de *n* tarefas ($J = \{J_1, J_2, ..., J_i, ..., J_n\}$) que devem ser processadas, na mesma seqüência, por um conjunto de *m* máquinas distintas, onde o tempo de processamento da tarefa J_i na máquina *k* é $p_{k,i}$ ($k = 1,2,...,m;\ i = 1,2,...,n$).

A principal característica do *NWFS* é a necessidade de que a operação $p_{g+1,i}$ de uma determinada tarefa J_i tem que ser processada logo após o término da operação $p_{g,i}$ $(1 \le g \le m-1)$, não permitindo que ocorra tempo de espera no processamento de uma tarefa de uma máquina para a outra (*no-wait*). O único tempo de espera permitido é no início do processamento das tarefas na primeira máquina. Este problema é classificado como $F / prmu, no\text{-}wait / C_{max}$ de acordo com a notação de GRAHAM *et al*. (1979).

O problema $F / prmu, no\text{-}wait / C_{max}$ também é considerado como *NP-Hard* para o caso de três ou mais máquinas (PAPADIMITRIOU *et al*., 1980). Entretanto, a pesquisa nesta área tem-se direcionado para o desenvolvimento de métodos heurísticos capaz de obter soluções de boa qualidade – embora não necessariamente ótimas – com relativo pequeno tempo computacional. Para o critério de minimização do *makespan*, REDDI e RAMAMOORTHY (1972) e WISMER (1972) foram os primeiros a estudar o problema. BONNEY e GUNDRY (1976), KING e SPACHIS (1980), foram os primeiros a desenvolver

as heurísticas para o problema. Finalmente, GANGADHARAN e RAJENDRAN (1993) e RAJENDRAN (1994) desenvolveram mais recentemente as heurísticas que apresentaram melhores desempenhos que as de BONNEY e GUNDRY (1976) e KING e SPACHIS (1980).

O objetivo deste trabalho é obter e avaliar o melhor método heurístico construtivo para o problema, pela combinação de diferentes procedimentos de ordenações e re-seqüenciamentos das tarefas. Nesse sentido uma extensa experimentação computacional é realizada de forma a avaliar o desempenho quanto a qualidade da solução e esforço computacional.

Finalizando, o melhor método encontrado é avaliado verificando o seu desempenho, resgatando as características essenciais de um método heurístico: adequado equilíbrio entre a qualidade da solução e esforço computacional, simplicidade e facilidade de implementação.

2. METODOS HEURÍSTICOS PARA A MINIMIZAÇÃO DA DURAÇÃO TOTAL DA PROGRAMAÇÃO

Os primeiros trabalhos para o problema *NWFS* para a minimização do *makespan* começaram a partir de BONNEY e GUNDRY (1976), que utilizaram relações geométricas e extensões de algoritmos para problemas de máquina única.

KING e SPACHIS (1980) apresentaram em sua pesquisa formas diferenciadas de ordenações de tarefas, comparando e avaliando os resultados quanto ao porte dos problemas gerados.

Nas últimas três décadas, um extenso esforço de pesquisa tem sido dedicado para o problema, e alguns métodos já propostos tem sido adaptados ou modificados para a solução de outros problemas, como, por exemplo, a heurística NEH (NAWAZ *et al*., 1983). Devido a sua eficiência e eficácia (qualidade da solução e tempo de computação), vários outros métodos têm-se baseado na estrutura da heurística construtiva NEH para o desenvolvimento de novos métodos em outros problemas de *Flow Shop* (GANGADHARAN e RAJENDRAN,1993; RAJENDRAN, 1994; FRAMINAN *et al*., 2003). A heurística NEH é composta de duas fases: a primeira de indexação das tarefas, onde as tarefas são ordenadas de acordo com os valores não-crescentes das somas dos tempos de processamentos, e a segunda de construção da seqüência solução. Nesse caso uma parcial seqüência é construída selecionando a primeira a tarefa obtida na primeira fase, em seguida, para $i = 2,...,n$, i seqüências parciais são construídas inserindo a i-ésima tarefa da ordenação da primeira fase,

ISSN 1676 - 1901 / Vol. X/ Num. X/ mês de ano

para todas as *i* possibilidades de acrescentar a tarefa na seqüência até então obtida, adota-se aquela que leva a uma menor duração total da programação parcial.

GANGADHARAN e RAJENDRAN (1993) desenvolveram dois métodos heurísticos, que comprovaram serem melhores que as heurísticas desenvolvidas anteriormente. Os métodos eram compostos de duas fases: uma primeira fase de geração e construção da seqüência, e uma segunda fase de melhoria da seqüência obtida na fase anterior.

RAJENDRAN (1994) apresentou um método heurístico considerado o melhor até o presente momento para o problema. A heurística de Rajendran consiste de uma fase indexação seguida por uma fase de construção da seqüência solução. A fase de indexação consiste primeiro no desenvolvimento de um índice P_i para cada tarefas ($P_i = \sum_{k=1}^{m} j \cdot p_{k\,i} \Big/ \sum_{k=1}^{m} p_{k\,i}$) onde, as tarefas são designadas em dois conjuntos *A* e *B* e dependendo dos valores de P_i serem maiores ou não que $(1+m)/2$. O conjunto *A* (*B*) é então ordenando de acordo com o valor não-decrescente (não-crescente) do índice $T_i = \sum_{k=1}^{m}(m-k+1)\cdot p_{k\,i}$, e ambos os conjuntos são agrupados. Na fase de construção da seqüência solução, a seqüência obtida na fase anterior é utilizada como semente para o processo de inserção, semelhante ao método de construção da seqüência solução do NEH (NAWAZ *et al*., 1983), onde neste caso cada tarefa *i* é inserido dentro da segunda metade da seqüência parcial, isto é, entre a $(k+1)/2$-ésima e $(k+1)$-ésima posição.

Outros pesquisadores têm direcionado seus esforços para a solução do problema utilizando-se de técnicas denominadas metaheurísticas (Algoritmos Genéticos, Busca Tabu, *Simulated Annealing*, *Ant Colony*), por exemplo, ALDOWAISAN e ALLAHVERDI (2003) que desenvolveram três heurísticas denominadas SA, SA-1 e SA-2, baseadas no método *Simulated Annealing* e outras três denominadas GA, GA-1 e GA-2 baseadas no método Algoritmo Genético. Os resultados computacionais apresentaram que significativas melhorias são obtidas se a heurística NEH é aplicada nos procedimentos metaheurísticos.

SCHUSTER e FRAMINAN (2003) apresentaram dois métodos de busca local, onde o primeiro foi baseado na variação do método de busca na vizinhança denominado *VNS* (*Variable Neighborhood Search*), o segundo foi um método hibrido *Simulated Annealing e* Algoritmo Genético denominado GASA. Os resultados analisados apresentaram que os dois métodos propostos obtêm resultados melhores que a heurística proposta por RAJENDRAN (1994).

Recentemente GRABOWSKI e PEMPERA (2005) apresentaram variações de algoritmos de busca descendente e Busca Tabu. Para os métodos propostos são usados multi-movimentos de forma a acelerar a convergência para boas soluções. Os resultados computacionais confirmaram que os métodos *DS* (*Descending Search*) são viáveis e confiáveis desde que não sejam complicados, e possam ser facilmente implementados computacionalmente. O melhor desempenho entre os métodos avaliados foi o *DS+M* (*Descending Search Algorithm with Multimoves*). Ao final da pesquisa os autores sugerem a diversificação do método para o problema *NWFS* com diferentes critérios de avaliação.

3. MÉTODOS HEURÍSTICOS

Os métodos heurísticos a serem avaliados nesta pesquisa são compostos de duas fases, semelhante ao melhor método heurístico construtivo proposto por Rajendran (1994) para o problema de *NWFS*.

Como foi apresentado anteriormente, uma extensa experimentação computacional será realizada de forma a obter um novo método heurístico pela combinação dos procedimentos de ordenação e de re-seqüenciamento das tarefas.

Incialmente foram analisados os principais métodos desenvolvidos apresentados na literatura para o problema *flow shop*, verificando as estruturas de composição dos métodos existentes, semelhante a pesquisa realizada por Framinan *et al.* (2003).

Após a análise dos principais métodos foram destacados oito procedimentos de ordenação inicial das tarefas e cinco procedimentos de re-seqüenciamento das tarefas conforme apresentado a seguir.

3.1. Procedimentos de ordenações iniciais das tarefas

NEH: Este procedimento de ordenação é semelhante a do método NEH (Nawaz *et al.*, 1983) para o problema de minimização do *makespan* em um *flow shop* tradicionais. O procedimento de ordenação é composto por dois passos:

Passo 1

Calcule, para cada tarefa, a soma dos tempos de processamento em todos as máquinas;

Passo 2

Ordene as *n* tarefas de acordo com os valores não-crescentes das somas dos tempos de processamento.

RAJ: Procedimento de ordenação semelhante a do método de Rajendran (1994), conforme apresentado anteriormente;

ALEAT: Procedimento de ordenação onde uma seqüência de tarefas é obtida por geração aleatoriamente;

BN1: Este procedimento de ordenação é baseado na expressão que fornece o tempo total da programação de pares de tarefas adjacentes, em quaisquer posições na seqüência, na condição de *no-wait* (Lawler *et al*., 1993). Para o caso de *m* máquinas uma expressão recursiva é apresentada, onde para todos os pares de tarefas J_i e J_j é obtido a duração total da programação pela expressão:

$$C^m_{max}(J_i, J_j) = p_{1i} + R_m(J_i, J_j) \quad (1)$$

para $i = 1, 2, ..., n$ e $j = 1, 2, ..., n$

Onde:

$$R_0(J_i, J_j) = 0$$

$$R_m(J_i, J_j) = p_{mj} + max(R_{m-1}(J_i, J_j); \sum_{k=2}^{m} p_{ki})$$

O procedimento de ordenação é formalmente descritos pelos seguintes passos, onde inicialmente são considerados: um conjunto J ($J = \{J_1, J_2, ..., J_n\}$) de n tarefas a serem programadas, S uma seqüência parcial de tarefas programadas, $S_{[k]}$ a posição k-ésima tarefa na seqüência S:

Passo 1 (Inicialização)

$J = \{J_1, J_2, J_3, ..., J_i, ..., J_n\}$;

$S = \emptyset$;

Selecione o maior elemento $C^m_{max}(J_i, J_j)$;

$S \leftarrow (J_j, J_i)$;

$J \leftarrow J - \{J_i, J_j\}$;

$u \leftarrow S_{[2]}$;

$k = 3$.

Passo 2

Selecione o menor elemento $C^m_{max}(J_u, J_v)$, tal que $J_v \in J$;

Examine todas as possibilidades de inserir a tarefa J_v na seqüência parcial (S), e adote a seqüência que leva a menor duração total da programação;

$J \leftarrow J - \{J_v\}$.

Passo 3

$u \leftarrow S_{[k]}$;

$k \leftarrow k + 1$;

Se $J \neq \emptyset$, volte para o *Passo 2*. Caso contrário, a ordenação inicial das tarefas está concluída.

Para as ordenações RESD1, RESD2 e RESD3 o mesmo procedimento de ordenação do BN1 é aplicado, diferenciando-se somente pelas expressões propostas por Stinson e Smith (1982). Nesse caso as expressões 2, 3 e 4 apresentam uma matriz de distância, onde tal problema transformado é análogo ao problema do Caixeiro-Viajante (*TSP – Traveling Selesman Problem*).

RESD1:

$$C(J_i, J_j) = \sum_{k=2}^{m} max(r_{k,ij}, 0) + 2\left| min(r_{k,ij}, 0)\right| \quad (2)$$

Onde: $r_{k,ij} = p_{ki} - p_{(k-1)j}$

RESD2:

$$C(J_i, J_j) = \sum_{k=2}^{m} \left(r^*_{k,ij}\right)^2 \quad (3)$$

Onde: $r^*_{k,ij} = p_{ki} - p_{(k-1)j} + min(r_{(k-1),ij}, 0)$

RESD3:

$$C(J_i, J_j) = \sum_{k=2}^{m} \left| r^*_{k,ij} \right| \quad (4)$$

Onde: $r^*_{k,ij} = p_{ki} - p_{(k-1)j} + min(r_{(k-1),ij}, 0)$

BN2: Este procedimento de ordenação utiliza a expressão 1, porém o procedimento se diferencia pelo seu esforço computacional na intensificação no procedimento de busca de uma melhor solução parcial. O procedimento apresenta os seguintes passos:

Passo 1 (Inicialização)

$J = \{J_1, J_2, J_3, ..., J_i, ..., J_n\}$;

$S = \emptyset$;

ISSN 1676 - 1901 / Vol. X/ Num. X/ mês de ano

Selecione o maior elemento $C_{max}^{m}(J_i, J_j)$;

$S \leftarrow (J_j, J_i)$;

$J \leftarrow J - \{J_i, J_j\}$;

$u \leftarrow S_{[2]}$;

$k = 3$.

Passo 2

Selecione o menor elemento $C_{max}^{m}(J_u, J_v)$ tal que $J_v \in J$;

Examine todas as possibilidades de inserir a tarefa J_v na seqüência parcial (*S*), e adote aquela que leva a menor duração total da programação;

$J \leftarrow J - \{J_v\}$;

Passo 3

Considerando a Vizinhança de Inserção examine todas as seqüências vizinhas obtidas com a tarefa J_v, e atualize a seqüência *S* com aquela que leva a menor duração total da programação;

Passo 4

Considerando toda a Vizinhança de Inserção da seqüência parcial com *k* tarefas, constituídas de $(k-1)^2$ seqüências, determine a seqüência S' associada a menor duração total da programação. Dada uma seqüência de tarefas, uma outra seqüência pertencente à sua vizinhança de inserção é obtida escolhendo-se uma das tarefas e uma das posições, inserido nessa posição a tarefa escolhida;

Considerando toda a Vizinhança de Permutação da seqüência parcial S' com *k* tarefas, constituída de $k(k-1)/2$ seqüências, determine a seqüência S'' associada a menor duração total da programação. Dada uma seqüência de tarefas, uma outra seqüência pertencente à sua vizinhança de permutação é obtida trocando-se as posições de duas tarefas quaisquer;

Atualize, com a seqüência S'', as *k* primeiras posições da seqüência de S';

$u \leftarrow S_{[k]}$;

$k \leftarrow k + 1$;

Se $J \neq \emptyset$, volte para o *Passo 2*. Caso contrário, a ordenação inicial das tarefas está concluída.

3.2. Procedimentos de re-seqüenciamentos das tarefas

Para a obtenção da seqüência final os procedimentos de re-seqüenciamento utilizados são os seguintes:

NEH: Este procedimento de re-seqüenciamento é semelhante ao NEH de Nawaz *et al.* (1983), e é composto por dois passos:

Passo 1

Selecione as duas primeiras tarefas da ordenação, seqüenciando-as de maneira a minimizar a duração total da programação, considerando-se somente essas duas tarefas;

Passo 2

Para $k = 3$ a n**,** faça;

Selecione a tarefa que ocupa a k-ésima posição na ordenação obtida no processo de ordenação;

Examine as k possibilidades de acrescentar a tarefa na seqüência até então obtida, adotando aquela que leva a uma menor duração total da programação parcial.

RAJ: Este procedimento de re-seqüenciamento é igual ao método proposto por Rajendran (1994), conforme apresentado anteriormente;

FL: Este procedimento de re-seqüenciamento proposto por Framinan e Leisten (2003) é apresentado a seguir:

Passo

Para $k = 3$ a n, faça;

Selecione as $k-1$ primeiras tarefas da seqüência inicial S. Examine todas as possibilidades de inserir a tarefa $S_{[k]}$ na seqüência parcial até então obtida, e adote aquela que leva a menor duração total da programação;

Considerando toda a Vizinhança de Permutação da seqüência parcial com k tarefas, constituídas de $(k-1)/k$ seqüências, determine a seqüência associada a menor duração total da programação.

BN: Este procedimento de re-seqüenciamento é uma modificação do método proposto por Framinan e Leisten (2003). A diferença fundamental está na forma de busca de seqüências vizinhas onde, no método FL, é utilizada a vizinhança de permutação de pares de tarefas. Para o procedimento proposto será utilizada a vizinhança de inserção de tarefa. A troca de um procedimento de busca na vizinhança por um outro pode aumentar o número de seqüências

avaliadas, permitindo desse modo uma maior possibilidade de soluções. A seguir é apresentado o procedimento de re-seqüenciamento proposto:

Passo

Para $k = 3$ a n, faça;

Selecione as $k-1$ primeiras tarefas da seqüência inicial S. Examine todas as possibilidades de inserir a tarefa $S_{[k]}$ na seqüência parcial até então obtida, e adote aquela que leva a menor duração total da programação;

Considerando toda a Vizinhança de Inserção da seqüência parcial com k tarefas, constituídas de $(k-1)^2$ seqüências, determine a seqüência associada a menor duração total da programação.

NM: Este procedimento de re-seqüenciamento tem como base o método proposto por Nagano e Moccellin (2005) adaptado para o problema de minimização da duração total da programação. O método consiste de um procedimento de intensificação da busca de uma melhor seqüência vizinha através da combinação de duas vizinhanças (permutação e inserção). O procedimento adaptado para o presente problema é composto pelos seguintes passos:

Passo 1

Selecione as duas primeiras tarefas da ordenação obtida no processo de ordenação, seqüenciando-as de maneira a minimizar a duração total da programação, considerando-se somente essas duas tarefas.

Passo 2

Para $k = 3$ a n, faça;

Selecione a tarefa que ocupa a k-ésima posição na ordenação obtida no processo de ordenação;

Examine as k possibilidades de inserir a tarefa na seqüência parcial até então obtida, adotando aquela que leva a menor duração total da programação parcial.

Passo 3

Armazene em S a melhor seqüência obtida no Passo 2. No final do processo iterativo, a solução do problema será a seqüência S.

Passo 4

Para $k = 3$ a n, faça;

Selecione as k primeiras tarefas da seqüência S e calcule a sua respectiva duração total da programação;

ISSN 1676 - 1901 / Vol. X/ Num. X/ mês de ano

Considerando toda a Vizinhança de Inserção da seqüência parcial com k tarefas, constituída de $(k-1)^2$ seqüências, determine a seqüência S' associada a menor duração total da programação. Dada uma seqüência de tarefas, uma outra seqüência pertencente à sua vizinhança de inserção é obtida escolhendo-se uma das tarefas e uma das posições, inserindo nessa posição a tarefa escolhida.

A seguir, considerando toda a Vizinhança de Permutação da seqüência parcial S' com k tarefas, constituída de $k(k-1)/2$ seqüências, determine a seqüência S'' associada a menor duração total da programação. Dada uma seqüência de tarefas, uma outra seqüência pertencente à sua vizinhança de permutação é obtida trocando-se as posições de duas tarefas quaisquer.

Atualize, com a seqüência S'', as k primeiras posições da seqüência solução S.

4. EXPERIMENTAÇÃO COMPUTACIONAL E ANÁLISE DOS RESULTADOS

Para a comparação dos métodos foi utilizada uma amostra constituída de 7200 problemas-teste, referente ao trabalho de Nagano *et al*. (2005). Nessa amostra, o número de tarefas $(n) \in \{5,6,7,8,9,10,20,30,40,50,60,70,80,90,100,110,120,130\}$, e o número de máquinas $(m) \in \{5,10,15,20\}$, com 100 problemas para cada combinação $(m \times n)$.

Os problemas foram agrupados seguindo as seguintes categorias:

- Pequeno porte: 2400 problemas:
  - tarefas $(n) \in \{5,6,7,8,9,10\}$ e maquinas $(m) \in \{5,10,15,20\}$;
- Médio porte: 2800 problemas:
  - tarefa $(n) \in \{20,30,40,50,60,70,80\}$ e máquinas $(m) \in \{5,10,15,20\}$;
- Grande porte: 2000 problemas:
  - tarefa $(n) \in \{90,100,110,120,130\}$ e máquinas $(m) \in \{5,10,15,20\}$;

Os tempos de processamento das tarefas foram gerados aleatoriamente no intervalo de variação 1 a 99, conforme uma distribuição uniforme.

Os métodos heurísticos desenvolvidos foram codificados em linguagem *DELPHI* e processados em um microcomputador *Pentium IV 3.00 GHz*.

As estatísticas usadas para avaliar o desempenho dos métodos foram a Porcentagem de Sucesso, o Desvio Médio Relativo e o Tempo Médio de Computação.

A primeira é definida pelo quociente entre o número de problemas para os quais o método obteve a melhor solução (*makespan*) e o número total de programas resolvidos. Obviamente, quando os métodos obtêm a melhor solução para um mesmo problema, suas Porcentagens de Sucesso são simultaneamente melhoradas.

O Desvio Relativo (DR_h) quantifica o desvio que o método h obtém em relação ao melhor *makespan* obtido para o mesmo problema, sendo calculado conforme segue:

$$DR_h(\%) = \frac{D_h - D^*}{D^*} \times 100 \qquad (5)$$

onde,

D_h é a duração total da programação (*makespan*) obtido pelo método h;

D^* é o melhor *makespan* obtido pelo(s) método(s), para um determinado problema.

O Tempo Médio de Computação de um método é calculado pela soma dos tempos de computação de cada problema dividida pelo número total de problemas resolvidos (média aritmética dos tempos de computação). Na experimentação computacional, o tempo médio de computação para o problema foi medido em milisegundo (*ms*).

Para os resultados a serem apresentados a seguir serão adotadas as seguintes notações:

* Porcentagem de Sucesso (%);

** Desvio Médio Relativo (%);

*** Tempo Médio de Computação (milisegundo).

4.1. Avaliação das ordenações iniciais

Primeiramente é apresentado o resultado dos oito procedimentos de ordenações iniciais (Tabela 1), agrupando os dados por número de máquinas e separando por categorias de problemas de pequeno, médio e grande porte.

Tabela 1 – Avaliação dos procedimentos de ordenações iniciais

	NEH	RAJ	BN1	ALEAT	RESD1	RESD2	RESD3	BN2
Pequeno Porte	0,21* 16,459** 1,00***	7,33 7,490 **0,91**	47,29 1,137 1,06	0,17 25,453 0,98	28,75 2,663 1,06	28,08 2,701 1,01	28,00 2,721 0,97	**79,67** **0,303** 1,02
Médio Porte	0,00 28,439 1,56	0,00 22,909 1,47	3,61 2,441 1,95	0,00 49,170 **1,42**	3,79 2,511 2,63	3,18 2,557 2,18	3,93 2,558 2,11	**86,18** **0,112** 16,78
Grande Porte	0,00 33,551 **2,83**	0,00 30,240 3,09	0,25 2,861 4,59	0,00 57,179 2,87	0,65 2,635 7,98	0,40 2,598 6,19	0,45 2,571 6,23	**98,30** **0,005** 184,62
Total	0,07 25,866 1,73	2,44 19,806 1,73	17,24 2,123 2,39	0,06 43,489 **1,68**	11,24 2,596 3,59	10,71 2,617 2,91	10,99 2,616 2,87	**87,38** **0,146** 58,14

A tabela 1 apresenta claramente a superioridade do procedimento de ordenação inicial BN2 comparado aos demais métodos. À medida que aumenta o número de tarefas, aumenta a porcentagem de sucesso do algoritmo BN2 em relação aos seus concorrentes. No total verificou-se que BN2 apresenta melhor ordenação em 6291 problemas, ou seja, em 87,38% dos problemas.

O desvio médio relativo tende a verificar os resultados anteriores. É conveniente lembrar que, quanto menor o valor do desvio médio relativo, maior é a qualidade da solução obtida pelo procedimento considerado. O procedimento BN2 também apresentou sua superioridade chegando muito próximo de desvio médio relativo zero em todas as faixas de problemas, principalmente para problemas de médio e grande porte.

Pode-se observar também na tabela 1 que o procedimento BN2 apresenta, em termos gerais, um desvio médio relativo muito menor que os demais procedimentos de ordenação inicial investigados. BN2 teve, na média dos 7200 problemas, um desvio médio relativo de 0,146%; o segundo menor desvio foi de 2,123% do procedimento BN1, aproximadamente 15 vezes maior.

Quanto ao tempo de computação, BN2 apresentou-se mais elevado, devido ao método de ordenação ser mais intensivo. Contudo, o tempo médio computacional não ultrapassa 0,2

segundo para problemas de grande porte, sendo assim aceitável e não relevante para fins de comparação.

4.2. Avaliação dos procedimentos de ordenações iniciais com os procedimentos de re-seqüenciamentos

Para a avaliação dos procedimentos de re-seqüenciamentos, todos os procedimentos de ordenações iniciais foram combinados com os procedimentos de re-seqüenciamentos. As tabelas abaixo apresentam os resultados obtidos em relação a porcentagem de sucesso, desvio médio relativo e tempo médio de computação.

Conforme apresentado nas tabelas abaixo, em todos os casos, a combinação que leva a ordenação inicial BN2 foi a que obteve a maior porcentagem de sucesso e menor desvio médio relativo. Os tempos médios de computação não ultrapassaram, na média, meio segundo e, portanto, nenhum método apresentou-se inviável por este critério.

Assim, podem-se destacar os cinco melhores métodos de duas fases, i.e., o melhor para cada procedimento de re-seqüenciamento. Analisando-se as tabelas 2A a 2E pode-se verificar que as combinações BN2+NEH, BN2+RAJ, BN2+FL, BN2+BN e BN2+NM foram as melhores. Os resultados da próxima experimentação terão como objetivo apresentar o melhor método de duas fases entre os 5 que mais se destacaram.

Tabela 2A – Avaliação dos métodos de duas fases

		Ordenações Iniciais							
		NEH	**RAJ**	**BN1**	**ALEAT**	**RESD1**	**RESD2**	**RESD3**	**BN2**
Re-seqüenciamentos NEH	**Pequeno Porte**	36,21*	35,46	55,75	36,17	47,46	48,38	48,21	**75,33**
		1,516**	1,628	0,703	1,591	0,992	0,978	1,010	**0,332**
		1,11***	**0,95**	1,06	1,05	1,10	1,00	1,06	1,15
	Médio Porte	2,18	3,18	4,96	3,07	5,75	5,21	6,25	**70,86**
		2,535	2,556	1,822	2,306	1,833	1,817	1,786	**0,223**
		2,19	2,15	2,53	**2,13**	3,30	2,81	2,91	17,03
	Grande Porte	0,35	0,30	0,80	0,50	1,45	1,30	1,80	**93,55**
		2,715	2,874	2,227	2,576	2,086	2,052	2,022	**0,021**
		7,25	7,94	9,80	**7,16**	12,85	11,52	10,90	188,58
	Total	13,01	13,14	20,74	13,39	18,46	18,51	19,00	**78,65**
		2,245	2,335	1,562	2,143	1,623	1,603	1,593	**0,203**
		3,24	3,36	4,06	**3,17**	5,22	4,63	4,51	59,39

Tabela 2B - Avaliação dos métodos de duas fases

		Ordenações Iniciais							
		NEH	**RAJ**	**BN1**	**ALEAT**	**RESD1**	**RESD2**	**RESD3**	**BN2**
Re-seqüenciamentos RAJ	**Pequeno Porte**	5,83*	26,75	55,54	14,04	36,21	36,13	35,33	**78,17**
		7,282**	2,342	0,783	5,325	2,010	2,123	2,124	**0,305**
		1,14***	0,98	**0,91**	1,02	1,07	1,07	1,00	1,05
	Médio Porte	0,00	0,93	4,11	0,00	4,39	3,54	4,43	**83,43**
		7,627	4,406	2,135	10,162	2,275	2,344	2,335	**0,130**
		2,44	2,12	2,12	**1,73**	3,00	2,62	2,89	17,20
	Grande Porte	0,00	0,00	0,35	0,00	0,75	0,45	0,55	**97,95**
		6,350	5,426	2,740	9,757	2,551	2,522	2,497	**0,007**
		6,50	6,90	9,32	**4,49**	13,02	10,60	10,72	188,21
	Total	1,94	9,28	20,21	4,68	13,99	13,54	13,65	**85,71**
		7,157	4,001	1,852	8,437	2,263	2,320	2,310	**0,154**
		3,13	3,07	3,72	**2,26**	5,14	4,32	4,44	59,32

Tabela 2C - Avaliação dos métodos de duas fases

		Ordenações Iniciais							
		NEH	**RAJ**	**BN1**	**ALEAT**	**RESD1**	**RESD2**	**RESD3**	**BN2**
Re-seqüenciamentos FL	**Pequeno Porte**	54,83*	49,79	63,83	52,63	58,13	60,42	58,79	**77,08**
		0,764**	0,959	0,486	0,869	0,644	0,623	0,644	**0,262**
		0,94* **	0,95	0,99	1,00	1,03	1,04	1,06	1,04
	Médio Porte	6,86	7,11	7,96	7,89	9,25	8,86	9,86	**45,32**
		1,458	1,552	1,319	1,426	1,318	1,312	1,292	**0,434**
		8,95	8,96	9,82	8,97	10,51	10,56	9,75	26,57
	Grande Porte	4,75	2,90	5,15	4,05	5,65	5,40	4,60	**68,10**
		1,211	1,373	1,211	1,334	1,171	1,160	1,164	**0,143**
		93,65	93,90	99,14	96,11	100,42	101,45	97,71	293,90
	Total	22,26	20,17	25,81	21,74	24,54	25,08	24,71	**62,24**
		1,158	1,305	1,011	1,215	1,053	1,040	1,040	**0,295**
		29,81	29,88	31,69	30,52	32,32	32,63	31,29	92,32

Tabela 2D - Avaliação dos métodos de duas fases

		Ordenações Iniciais							
		NEH	**RAJ**	**BN1**	**ALEAT**	**RESD1**	**RESD2**	**RESD3**	**BN2**
Re-seqüenciamentos BN	**Pequeno Porte**	58,29*	48,42	62,83	52,00	58,25	59,38	58,38	**77,50**
		0,733**	1,136	0,554	0,960	0,699	0,682	0,710	**0,286**
		2,58***	1,59	1,59	1,66	**1,57**	1,58	2,60	1,08
	Médio Porte	10,75	6,18	9,93	9,71	9,29	9,64	10,82	**37,07**
		1,212	1,555	1,289	1,323	1,261	1,258	1,238	**0,537**
		13,23	**10,79**	11,30	11,07	12,27	11,86	13,64	27,35
	Grande Porte	10,25	3,70	7,25	6,65	6,70	8,10	9,00	**49,40**
		0,990	1,342	1,115	1,161	1,082	1,030	1,039	**0,268**
		109,18	**104,91**	107,81	105,26	110,71	111,62	112,57	306,33
	Total	26,46	19,57	26,82	22,96	24,89	25,79	26,17	**53,97**
		0,991	1,356	0,996	1,157	1,024	1,003	1,007	**0,378**
		36,33	**33,87**	34,87	34,10	36,05	36,15	37,44	96,09

Tabela 2E - Avaliação dos métodos de duas fases

		Ordenações Iniciais							
		NEH	**RAJ**	**BN1**	**ALEAT**	**RESD1**	**RESD2**	**RESD3**	**BN2**
Re-seqüenciamentos NM	**Pequeno Porte**	57,75*	49,83	63,54	54,42	59,88	60,75	60,08	**76,71**
		0,717**	1,057	0,499	0,858	0,640	0,629	0,638	**0,266**
		1,02***	**0,94**	1,05	0,97	1,00	1,04	1,08	1,08
	Médio Porte	4,82	4,32	6,00	5,11	7,89	7,29	7,71	**59,00**
		1,914	2,041	1,575	1,804	1,549	1,522	1,511	**0,318**
		17,53	**17,31**	17,85	17,86	19,04	18,31	18,41	35,27
	Grande Porte	0,85	1,05	1,90	1,35	2,70	2,45	2,40	**87,70**
		2,294	2,478	1,951	2,195	1,837	1,802	1,781	**0,045**
		192,01	197,77	195,02	192,08	198,79	195,50	197,20	404,06
	Total	21,36	18,58	24,04	20,50	23,78	23,76	23,69	**72,88**
		1,621	1,834	1,321	1,597	1,326	1,302	1,295	**0,225**
		60,49	61,98	61,46	60,62	62,96	61,77	62,30	126,31

A tabela 2B apresenta que o principal método da literatura RAJ+RAJ (Rajendran, 1994), foi superado pelo método composto BN2+RAJ. Desta forma para a avaliação final dos melhores métodos de duas fases não será considerado o método RAJ+RAJ (Rajendran, 1994), já que o método proposto (BN2+RAJ) é superior.

4.3. Avaliação entre os melhores métodos de duas fases

A tabela 3 apresenta os resultados finais da experimentação computacional para os cinco melhores métodos avaliados.

Tabela 3 - Avaliação dos melhores métodos de duas fases

		Re-seqüenciamentos				
		NEH	**RAJ**	**FL**	**BN**	**NM**
Ordenação Inicial BN2	**Pequeno Porte**	84,50*	80,21	**92,21**	91,79	91,83
		0,202**	0,296	0,100	0,109	**0,083**
		1,15***	1,05	**1,04**	1,08	1,08
	Médio Porte	52,93	52,64	67,82	**78,82**	59,46
		0,370	0,429	0,215	**0,123**	0,283
		17,03	17,20	26,57	27,35	35,27
	Grande Porte	62,55	62,55	71,10	**90,10**	64,05
		0,198	0,205	0,137	**0,032**	0,186
		188,58	**188,21**	293,90	306,33	404,06
	Total	66,13	64,58	76,86	**86,28**	71,53
		0,266	0,323	0,155	**0,093**	0,190
		59,39	**59,32**	92,32	96,09	126,31

Para o problema de pequeno porte, pode-se perceber que são três os principais métodos com a maior porcentagem de sucesso: BN2+BN, BN2+NM e BN2+FL. À medida

que aumenta o número de tarefas, BN2+BN começa a se destacar, distanciando-se dos demais métodos, a partir dos problemas de médio porte. Os métodos BN2+NEH e BN2+RAJ apresentaram praticamente o mesmo resultado, evidenciando que os re-seqüenciamentos NEH e RAJ não são determinantes para uma melhoria significativa das soluções. Para o problema de grande porte, o comportamento é semelhante ao problema de médio porte, onde se destaca BN2+BN. De forma geral o melhor desempenho foi do algoritmo BN2+BN para os 7200 problemas analisados, o número de sucessos foi de 6212 problemas, ou 86,28%.

A tabela 3 vem comprovar a qualidade da solução do método BN2+BN como o melhor método através do desvio médio relativo. BN2+BN teve um desvio médio relativo de 0,093% em 7200 problemas, cerca de 40% menor que o desvio do segundo melhor algoritmo, BN2+FL.

5. CONSIDERAÇÕES FINAIS

Neste trabalho foi apresentado e avaliado um novo método heurístico construtivo, denominado BN2+BN, para o problema *NWFS* com o objetivo de minimizar a duração total da programação (*makespan*).

Preliminarmente, ressalta-se que para fins práticos, as soluções obtidas pelos métodos heurísticos de duas fases já existentes, para o problema *NWFS*, são suficientes, ou seja, tal problema pode ser considerado como já resolvido.

Porém, tendo em vista a complexidade do problema em questão, a busca por novos métodos que contêm adequado equilíbrio entre a qualidade da solução e a eficiência computacional, simplicidade e facilidade de implementação, ainda continua como uma direção para novas pesquisas.

O principal aspecto a ser destacado neste artigo refere-se ao fato de que os resultados experimentais mostraram que o método heurístico proposto BN2+BN possui um desempenho superior aos demais, para solução do problema em questão.

Dessa forma, este trabalho apresentou uma contribuição que procura evidenciar que apesar dos algoritmos existentes proporcionarem boas soluções, é possível, por meio da propriedade já existente (Lawler, 1993), criar novos métodos de soluções para o problema.

REFERENCIAS

ALDOWAISAN, T.; ALLAHVERDI, A. **New heuristics for no-wait flowshops to minimize makespan**. *Computers and Operations Research*, v.30, p.1219-1231, 2003.

BONNEY, M. C.; GUNDRY S. W. **Solutions to the constrained flowshop sequencing problem**. *Operations Research Quarterly*, v.24, p.869-883, 1976.

FRAMINAN, J. M.; LEISTEN R. **An efficient constructive heuristic for flowtime minimisation in permutation flow shops**. *Omega - The International Journal of Management Science*, v.31, n.4, p.311-317, 2003.

FRAMINAN, J. M.; LEISTEN, R.; RAJENDRAN, C. **Different initial sequences for the heuristic of Nawaz, Enscore and Ham to minimize makespan, idletime or flowtime in the static permutation flowshop sequencing problem**. *International Journal of Production Research*, v.41, n.1, p.121-148, 2003.

GANGADHARAN, R.; RAJENDRAN, C. **Heuristic algorithms for scheduling in the no-wait flowshop**. *International Journal of Production Economics*, v.32, p.285-290, 1993.

GRABOWSKI, J.; PEMPERA, J. **Some local search algorithms for no-wait flow-shop problem with makespan criterion**. *Computer and Operations Research*, v.32, p.2197-2212, 2005.

HALL, N. G.; SRISKANDARAYAH, C. **A survey of machine scheduling problems with blocking and no-wait in process**. *Operations Research*, v.44, p.510-525, 1996.

KING, R.; SPACHIS, A. S. **Heuristics for flow shop scheduling**. *International Journal of Production Research*, v.18, p.343-357, 1980.

LAWLER, E. L.; LENSTRA, J. K.; RINNOOY KAN, A. H. G.; SHMOYS, D. B. **Sequencing and scheduling: algorithms and complexity**. In: Graves, S.C. e Rinnooy, 1993.

NAGANO, M. S.; MOCCELLIN, J. V. **Redução do estoque em processamento em sistemas de produção flowshop**. In: *XXXVII Simpósio Brasileiro de Pesquisa Operacional*, Gramado-RS. Anais do XXXVII SBPO. 37, CD-ROM, 2005.

NAGANO, M. S.; MOCCELLIN, J.V.; LORENA, L. A. N. **Redução do estoque em processamento em sistemas de produção flow shop permutacional**. *Revista produção on line*, Florianópolis - SC, 5,. v.3, p.1-12, 2005.

NAWAZ, M.; ENSCORE JR., E. E.; HAM, I. **A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem**. *Omega - The International Journal of Management Science,* v.11, n.1, p.91-95, 1983.

PAPADIMITRIOU, C.; KANELLAKIS, P. C. **Flowshop scheduling with limited temporary storage**. *Journal of the Association for Computing Machinery*, v.27, p.533-549, 1980.

RAJENDRAN, C. **A no-wait flowshop scheduling heuristic to minimize makespan**. *Journal of the Operational Research Society*, v.45, n.4, p.472-478, 1994.

RAAYMAKERS, W.; HOOGEVEEN, J. **Scheduling multipurpose batch process industries with no-wait restrictions by simulated annealing**. *European Journal of Operational Research*, v.126, p.131-151, 2000.

STINSON, J. P.; SMITH, W. **A heuristic programming procedure for sequencing the static flowshop**. *International Journal of Production Research*, v.20, p.753-764, 1982.

WISMER, D. A. **Solution of the flowshop-scheduling with no intermediate queues**. *Operations Research*, v.20, p.689-697, 1972.

Artigo recebido em 2007 e aceito para publicação em 2008