Minimização da duração total da programação em sistemas de produção *flowshop*, sem interrupção de execução e tarefas

Fábio José Ceron Branco (EESC-USP) – fbranco@hotmail.com
• R. General Glicério, 340, Centro, CEP 15900-000, Taquaritinga-SP
Marcelo Seido Nagano (EESC/USP) – drnagano@usp.br
João Vitor Moccellin (EESC/USP) – jvmoccel@sc.usp.br

Recebido em: 09/08/07 Avaliado em: 04/09/07

Resumo
Este artigo trata do problema de programação da produção em sistemas de produção flowshop sem interrupção na execução das tarefas, conhecido também como no-wait flowshop. A função-objetivo é a minimização do tempo total da programação (makespan). Um novo método heurístico é proposto e comparado com outros métodos reportados na literatura. Os resultados experimentais mostram a superioridade do novo método para o conjunto de problemas avaliados.
Palavras-chave: *métodos heurísticos, no-wait flowshop, duração total da programação.*

Abstract
This paper addresses the m-machine no-wait flow shop scheduling problem to minimize makespan. The new method is compared with other improvement strategies reported in the literature. Experimental results show that the new heuristic provides better solutions regarding both solution quality and computational effort.
Keywords: *heuristics methods, no-wait flowshop, Makespan.*

GEPROS. Gestão da Produção, Operações e Sistemas – Ano 3, nº 2, abr-jun/08, p. 39-47

1. INTRODUÇÃO

Com o desenvolvimento das indústrias manufatureiras, o advento dos métodos de produção just-in-time e a filosofia do inventário zero, a programação da produção ganhou grande importância, principalmente em sistema de produção *flowshop*.

O problema tradicional de programação em sistema de produção *flowshop* consiste de um conjunto de *n* tarefas que devem ser processados, 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 *flowshop* permutacional, no qual o número de possíveis programações para *n* tarefas é *n*!. Usualmente, o problema consiste em determinar a programação que minimiza (makespan) a duração total da programação.

Uma situação específica freqüente é a programação da produção em sistema de produção *flowshop*, sem interrupção na execução das tarefas. Geralmente ocorre em um ambiente caracterizado pela tecnologia do processo. Por exemplo, quando a variação de temperatura pode influenciar na degeneração do material ou quando a capacidade de armazenamento entre as máquinas é um fato limitante.

Outras situações são encontradas nas indústrias químicas, metalúrgicas, alimentícias, de serviços e farmacêuticas.

Em uma indústria de metais, como o processamento de alumínio e aço, a laminação de placas passa por uma série de processos. Primeiro, a placa é pré-aquecida e depois, conduzida para uma série de prensas, na qual sua espessura é diminuída progressivamente até atingir sua especificação final. A placa tem que passar de uma máquina à outra, na seqüência, sem ficar esperando pela disponibilidade da próxima máquina. Qualquer espera pode causar o resfriamento do metal, gerando problemas na laminação e, conseqüentemente, perda de material.

No caso de indústria química de um composto químico, depois de passar por um misturador, tem que ser envasado, logo em seguida, pois a exposição ao ar pode deteriorá-lo. Outros exemplos que podem ser citados referem-se à indústria de alimentação, onde o alimento tem que ser enlatado após seu cozimento para garantir sua qualidade e ao setor de serviços, pois o cliente tem uma alta intolerância à espera.

Em tais ambientes apresentados procura-se reduzir custos com inventários e planejar a produção de forma a atender a demanda, sem que haja excessos ou falta do produto, maximizando a utilização dos recursos e barateando o processo, de acordo com a demanda do mercado.

Este problema também é chamado de *flowshop* sem espera ou *no-wait flowshop* (*NWFS*), conforme apresentado na figura 1.

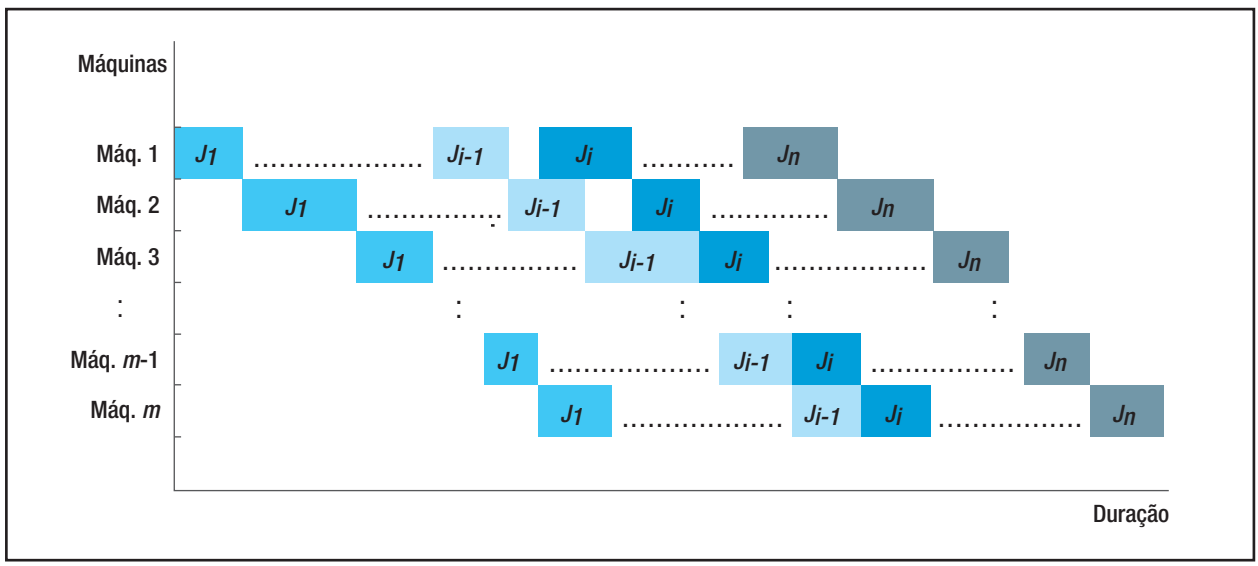

FIGURA 1 – *No-wait flowshop* com m máquinas e n tarefas.

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 t_{ik} ($i = 1,2,...,n$; $k = 1,2,...,m$).

A principal característica do *NWFS* é a necessidade de que a operação $g + 1$, de uma determinada tarefa J_i, tem que ser processada logo após o término da operação g ($1 \leq g \leq m - 1$), não permitindo que ocorra tempo de espera no processamento de uma tarefa de uma máquina para a outra. O único tempo de espera permitido é no início do processamento da tarefa, que ocupa a primeira posição na primeira máquina.

Devido à sua natureza, o problema *NWFS* é considerado como NP-Hard para o caso de três ou mais máquinas (PAPADIMITRIOU & KANELLAKIS, 1980).

O objetivo deste trabalho é apresentar e avaliar métodos heurísticos para o problema de programação de operações em ambientes *NWFS*, com o critério de minimização do makespan.

Um novo método heurístico é proposto e comparado com o melhor método existente na literatura, por meio de uma extensa experimentação computacional, verificando-se o seu desempenho.

Para o desenvolvimento do novo método heurístico, foram consideradas as seguintes características essenciais: adequado equilíbrio entre a qualidade da solução, eficiência computacional, simplicidade e facilidade de implementação.

2. O PROBLEMA *NO-WAIT FLOWSHOP*

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

King & Spachis (1980) apresentaram heurísticas com formas diferenciadas de ordenações de tarefas, comparando e avaliando os resultados quanto ao porte dos problemas gerados.

Gangadharan & Rajendran (1993), desenvolveram dois métodos heurísticos para minimização do makespan que se mostraram melhores que as heurísticas desenvolvidas por King & Spachis (1980). Ambos são compostos de duas fases: em uma primeira, os autores propõem uma ordenação inicial; em uma segunda, a solução final é construída.

Rajendran (1994), apresentou uma evolução de Bonney & Gundry (1976) e de King & Spachis (1980). O método é estruturalmente semelhante aos de Gangadharan & Rajendran (1993), pois também é composto por duas fases. Na primeira, a ordenação inicial é determinada por um índice calculado para cada tarefa e na segunda, utiliza um procedimento que avalia seqüências parciais até que todas as tarefas estejam programadas. O método construtivo de Rajendran (1994) é o melhor encontrado na literatura para o problema *NWFS*, com critério de minimização do makespan.

Nas últimas três décadas, um extenso esforço de pesquisa tem sido dedicado ao problema e alguns métodos propostos têm se baseado em outros existentes, como, por exemplo, o método heurístico NEH (NAWAZ et al., 1983). Devido ao seu desempenho (qualidade da solução e tempo de computação) vários pesquisadores têm proposto sua adaptação para os problemas de *flowshop* com restrições adicionais ou com diferentes critérios de avaliação (FRAMINAN et al., 2003).

Outros pesquisadores têm direcionado seus esforços para a solução do problema utilizando-se de técnicas denominadas meta-heurísticas (Algoritmos Genéticos, Busca Tabu, Simulated Annealing, Ant Colony) como, por exemplo, a pesquisa apresentada por Aldowaisan & Allahverdi (2003) que desenvolveram três heurísticas baseadas no método Simulated Annealing e outras três, baseadas no método Algoritmo Genético.

Neste trabalho apresentam-se os resultados de uma pesquisa com o desenvolvimento de um novo método heurístico para o problema *NWFS* comparando o desempenho com os métodos propostos por Rajendran (1994) e Nawaz et al. (1983), adaptados ao problema *NWFS*. O novo método proposto é baseado em uma propriedade específica do problema, conforme apresentado por Lawler et al. (1993).

3. MÉTODO HEURÍSTICO PROPOSTO

Os métodos heurísticos de Rajendran (1994) e de Nawaz et al. (1983) possuem duas fases: a primeira, um procedimento de ordenação das tarefas e, na posterior, um procedimento de re-seqüenciamento.

O método proposto para essa pesquisa é um método de uma fase que utiliza a propriedade desenvolvida por Lawler et al. (1993). Ela fornece o tempo total da programação de pares de tarefas J_i e J_j, ($C^m_{\max}(J_i,J_j)$) para o caso de m máquinas, onde p_{uv} é o tempo de processamento da tarefa v na máquina u, e R_m é o resíduo do tempo alocado na programação de pares de tarefas, na máquina m, conforme a 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:

$R_o(J_i,J_j) = 0$

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

Para o presente trabalho o método proposto será chamado de BN, sendo composto pelos seguintes passos:

Seja J um conjunto de n tarefas ($J = \{J_1,J_2,...,J_n\}$) a serem programadas, S o conjunto das tarefas programadas, $S_{[k]}$ a tarefa que ocupa a k-ésima posição de S.

Passo 1 (inicialização)

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

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 aquela que leva a menor duração total da programação:

$S \leftarrow S \cup \{S_{[v]}\}$;
$J \leftarrow J - \{J_v\}$;

Passo 3

Considerando toda a Vizinhança de Inserção da seqüência parcial com k tarefas constituídas de $(k-1)^2$ seqüências, determine a seqüência S', associada à 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 nesta posição a tarefa escolhida;

Atualize com a seqüência S', as k primeiras posições da seqüência de S, apenas se a duração total da programação de S' for menor que a de S;

Passo 4

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 à 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, apenas se a duração total da programação de S'' for menor que a de S;

$u \leftarrow S_{[k]}$;
$k \leftarrow k + 1$;

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

4. EXPERIMENTAÇÃO COMPUTACIONAL E ANÁLISE DOS RESULTADOS

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

Conforme a variação do número de tarefas, os problemas foram classificados em:

Pequeno porte: 2400 problemas:
- tarefas: 5, 6, 7, 8, 9 e 10;
- máquinas: 5, 10, 15 e 20.

Médio porte: 2800 problemas:
- tarefa: 20, 30, 40, 50, 60, 70 e 80;
- máquinas: 5, 10, 15 e 20.

Grande porte: 2000 problemas:
- tarefa: 90, 100, 110, 120 e 130;
- máquinas: 5, 10, 15 e 20.

Os métodos heurísticos avaliados foram codificados em linguagem de programação Delphi e processados, conjuntamente, em um microcomputador Pentium IV 3.00 GHz. Todos os métodos implementados tiveram, como objetivo, visar a sua funcionalidade, apresentando de forma detalhada os resultados para cada problema, com sua respectiva seqüência-solução, valor da função-objetivo (makespan) e tempo de computação.

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

As estatísticas usadas para avaliar o desempenho dos métodos foram a Percentagem 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 um determinado 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 Percentagens 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:

$$DR_h\,(\%) = \frac{D_h - D^*}{D^*} \times 100 \qquad (2)$$

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 obtido pela soma dos tempos de computação de cada problema, dividida pelo número total de problemas resolvidos. Na experimentação computacional, o tempo médio de computação por problema foi medido em milissegundo (ms).

A tabela 1 apresenta os resultados dos algoritmos, BN, RAJ (RAJENDRAN, 1994) e NEH-*NWFS* (NAWAZ et al., 1983) adaptado para o problema *NWFS*.

TABELA 1 – Resultados globais entre os algoritmos.

Número de Tarefas	BN	NEH-*NWFS*	RAJ
5	86,75* 0,221** 0,81***	64,75 1,046 1,10	51,25 1,700 0,78
6	80,25 0,284 0,90	49,75 1,154 1,18	33,50 2,192 0,98
7	72,00 0,460 0,98	45,50 1,226 1,06	28,00 2,459 1,01
8	65,75 0,777 1,12	40,50 1,677 1,13	28,00 2,472 0,94
9	64,00 0,573 1,43	35,25 1,735 0,86	24,50 2,743 1,02

10	66,00 0,523 2,07	28,25 2,014 1,09	18,00 2,977 0,94
20	60,75 0,562 3,60	28,25 2,140 1,09	12,00 3,634 1,40
30	63,75 0,460 4,58	30,50 1,860 1,41	6,75 3,513 1,30
40	63,00 0,427 8,05	33,75 1,441 1,56	3,50 3,364 1,44
50	68,25 0,332 14,73	30,50 1,391 2,04	1,75 3,525 1,96
60	63,00 0,383 27,03	35,50 1,071 2,53	1,75 3,318 2,39
70	62,50 0,300 46,64	35,75 0,898 3,01	2,25 3,343 2,85
Número de Tarefas	BN	NEH-*NWFS*	RAJ
Número de Tarefas	BN	NEH-*NWFS*	RAJ
80	68,50 0,246 76,33	31,75 0,913 3,71	0,25 3,346 3,52
90	65,50 0,287 117,27	34,50 0,854 4,88	0,00 3,428 4,53
100	62,75 0,227 182,54	36,75 0,818 5,40	1,00 3,377 5,50
110	66,50 0,237 253,25	33,50 0,800 6,95	0,25 3,557 6,60
120	63,25 0,305 356,44	38,00 0,638 8,64	0,00 3,424 8,08
130	58,00 0,170 496,76	42,25 0,656 10,39	0,50 3,371 9,77

*Porcentagem de Sucesso (%)..

**Desvio Médio Relativo (%)..

***Tempo Médio Computacional (milisegundo).

Apesar de ser um algoritmo de apenas uma fase, o método BN apresentou a maior percentagem de sucesso quando comparado com os métodos RAJ e NEH-*NWFS*, ambos de duas fases. A tabela 1 apresenta claramente sua superioridade em todos os portes de problema. Verificou-se que BN apresentou a melhor solução em 4802 problemas de um total de 7200, ou seja, 66,70%.

Analisando-se os desvios médios relativos verificou-se que o método BN gera melhores resultados em relação à qualidade das soluções, pois seus desvios médios relativos são menores que os demais métodos investigados. Na média geral dos 7200 problemas, o desvio médio relativo foi de 0,376%.

Em relação ao tempo de computação o método BN apresentou maior tempo devido à forma intensiva de busca de melhores soluções na construção da seqüência solução. Contudo, o tempo médio de computação não ultrapassou 0,5 segundo para problema de grande porte, sendo, assim, aceitável e não relevante para fins de comparação.

Outra observação interessante foi o desempenho do método NEH-*NWFS* diante do método RAJ. O método NEH adaptado para o problema de no-wait (NEH-*NWFS*) produziu resultados melhores que o método de Rajendran (1994).

5. CONSIDERAÇÕES FINAIS

Neste trabalho foi apresentado e avaliado um novo método heurístico construtivo denominado BN, aplicável à no-wait, com o objetivo de minimizar a duração total da programação (makespan). O método BN é um método de fase única, de fácil implementação e foi comparado com os melhores métodos de duas fases: Rajendran (1994) e de Nawaz et al. (1983), adaptado ao problema de no-wait.

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

Tendo em vista, porém, a complexidade do problema em questão, a busca por novos métodos com 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 novo método heurístico proposto possui um desempenho superior aos demais para a solução do problema em questão. Desta forma, este trabalho apresenta a evidência de que, apesar dos algoritmos existentes proporcionarem boas soluções, é possível, por meio de propriedade já existente, criar novos métodos de soluções, obtendo desempenhos superiores aos já existentes.

6. REFERÊNCIAS BIBLIOGRÁFICAS

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.; 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.

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.; RINNOOY, 1993.

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**, v. 5, n. 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.