www.revistageintec.net
ISSN: 2237-0722
revista GEINTEC Gestão, Inovação e Tecnologias

SEQUENCIAMENTO DE TAREFAS COM TEMPOS DE SETUP VIA BRANCH AND BOUND
SEQUENCING OF TASKS WITH SETUP TIMES USING BRANCH AND BOUND

Edilson de Jesus[1], Maria Teresa Rodrigues[2]
[1]Universidade Federal de Sergipe – UFS – DEQ- São Cristóvão/SE – Brasil
edilsonjs@ufs.br
[2]Universidade Estadual de Campinas – UNICAMP – DEQ – Campinas/SP – Brasil
maite@feq.unicamp.br

Resumo

Este trabalho propõe limitante inferior a ser utilizado no procedimento de busca em árvore para a solução de sequenciamento de produção com tempos de setup (tempos de estabelecimento) em ambiente flowshop visando minimizar o tempo total de execução das tarefas. O problema de sequenciamento consiste em definir a melhor sequência, segundo critério de otimalidade predeterminado, em que N tarefas devem ser desenvolvidas em M máquinas. Neste trabalho, é proposta uma formulação para o cálculo do limitante inferior a ser utilizado na busca em árvore branch and bound para resolver o problema de sequenciamento de tarefas considerando N tarefas e M máquinas, envolvendo tempos de setup. O algoritmo de busca em árvore branch and bound foi implementado em Pascal. As relações de recorrências propostas levaram a obtenção da sequência mínima. Neste trabalho foi detalhado para 4 tarefas e 2 máquinas.

Palavras-chave: flowshop; tempos de estabelecimento; busca em árvore; Otimização.

Abstract

This paper proposes lower bound to be used in the tree search procedure for solving production scheduling with setup times flowshop environment in order to minimize the total execution time of tasks limiting. The sequencing problem is to define the best sequence according to predetermined criteria of optimality, where N tasks must be developed in M machines. In this work, a formulation for the calculation of the lower being used in the search tree branch and bound bound is proposed to solve the problem of task sequencing considering N tasks and M machines, involving setup times. The search algorithm on the tree branch and bound was implemented in Pascal. The recurrence relations proposals have led to obtaining the minimum sequence. This work was detailed for 4 jobs and 2 machines.

Key-words: flowshop, setup times, branch and bound, optimization.

1. Introdução

No desenvolvimento do planejamento e programação de produção, que é vital para a diminuição de custos de qualquer setor produtivo, é possível utilizar métodos de programação matemática, métodos de inteligência artificial, em especial o método branch and bound, e métodos heurísticos visando à otimização dos processo de produção (SANTOS, 1994 e 1998; GONÇALVES JÚNIOR e JESUS, 2012).

A utilização da programação matemática é bastante difundida em diversas áreas para a solução de problemas de Engenharia Química. Os modelos resultantes quando se aborda um determinado problema da engenharia têm sua dificuldade inerente à presença de variáveis inteiras e de restrições não lineares, a exemplo dos modelos que desenvolvidos para a solução de problemas de programação de produção, os quais, em regra, envolvem restrições não lineares e variáveis inteiras.

Um problema bastante estudado na literatura é o *flowshop*, em que N tarefas devem ser executadas na mesma sequência em um conjunto de M máquinas distintas. Um caso específico de problema *flowshop* é quando em cada máquina a ordem de execução das tarefas é a mesma (CHAVES *et al*., 2007), tendo-se, portanto, um problema de sequenciamento de tarefas, sem haver necessidade de dimensionar as quantidades (lotes) a serem produzidas.

O problema de sequenciamento de tarefas com sequência dependente do tempo de *setup* é de difícil solução e, em regra, remete ao problema do caixeiro viajante. Este problema foi formulado inicialmente por Egon Balas (BALAS, 1989) como um modelo para a programação da produção em indústria de lâminas de aço. Já Gomes *et al*. (2000) e Melo (2001) desenvolveram algoritmos híbridos para resolver extensões do problema do caixeiro viajante.

Neste trabalho, é proposta uma formulação para o cálculo do limitante inferior a ser utilizado na metodologia de busca em árvore *branch and bound* para resolver o problema de sequenciamento de tarefas considerando N tarefas e M máquinas, envolvendo tempos de *setup.* O detalhadamento do algoritmo proposto foi feito considerando 4 tarefas e 2 máquinas.

Esta busca é considerada uma ferramenta de enumeração implícita, em que há o controle de formação de problemas parciais pela função objetivo. O artigo está organizado da seguinte forma: na seção 2 é apresentada breve descrição do problema de programação de produção em plantas flexíveis (possuem várias rotas de produção); na seção 3 é apresentada a aplicação do método *branch and bound* em problemas de sequenciamento de tarefas; na seção 4 são apresentadas as relações de recorrência para a solução do problema de sequenciamento com *setup*; na seção 5 são apresentados os resultados e discussões, e, por último, na seção 6 é feita a conclusão.

2. Programação de produção em sistemas flexíveis

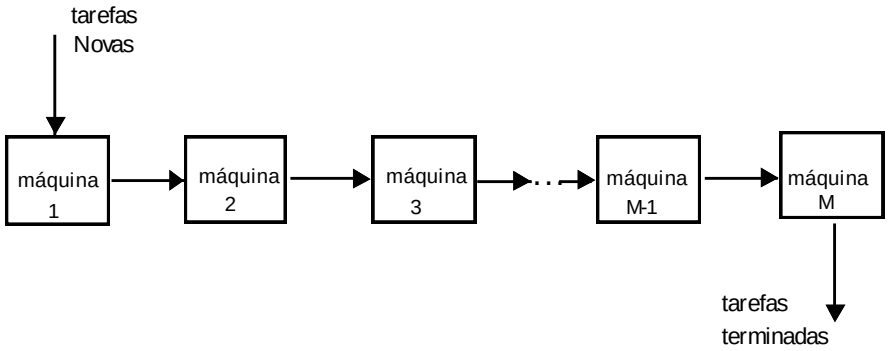
O processamento contínuo de matérias-primas é mais desejável que o modo não contínuo (processo em batelada). No entanto, na indústria química é comum a existência destes processos, a exemplo da produção de tintas, de fármacos e de tingimento de tecidos.

Vários trabalhos de revisão abordam o modo de processamento em batelada (REKLAITIS,1982 e 1992; KU *et al*, 1987). No que se refere à produção em batelada, pode-se dizer que é vital a programação de produção para o desenvolvimento das atividades inerentes a estes processos, visando à obtenção de um plano ótimo de produção.

Ku *et al*. (1987) relatam que a produtividade e eficácia dos processos uma indústria dependem de maneira crítica do roteiro de produção, sendo necessário que os recursos disponíveis sejam utilizados de modo racional durante a execução das diversas tarefas.

As indústrias são classificadas em plantas multiproduto e plantas multipropósito. Em plantas multiproduto as tarefas são executadas em uma mesma sequência de produção. A principal característica das plantas multiproduto é a relação de precedência linear entre os estágios de produção. Quando a planta batelada multiproduto é caracterizada pela ausência de semicontínuos a planta recebe o nome de *flowshop* puro, como mostrado na Figura 1, onde somente equipamentos operando em batelada são considerados.

Figura 1: Planta *flowshop*



Em plantas multipropósito um conjunto de diferentes operações pode estar presente ao mesmo tempo na planta e o mesmo produto pode ser processado por rotas diferentes.

Em plantas multiproduto, determinar um programa de produção ótimo envolve principalmente uma distribuição de tarefas no tempo. Em plantas multipropósito uma determinada tarefa pode ser executada em diferentes equipamentos, consequentemente a escolha da tarefa e do equipamento a ser utilizada deve ser feita previamente (SANTOS, 1994).

A procura dos setores industriais por processos batelada deve-se em parte pela grande flexibilidade de produção e natureza do produto. Hejazi e Saghafian (2005) apresentam uma ampla revisão da literatura no tocante a problemas de produção *flowshop*, mostrando que a maioria das pesquisas efetuadas considera os tempos de *setup* não significativos ou então os incluem nos tempos de processamento das tarefas. Para Moccellin *et al*. (2007) esta consideração simplifica a solução de problemas, porém limita a utilização do modelos em casos reais.

Chen (2009) propõe um modelo de programação de produção envolvendo somente uma máquina, com interrupção de trabalho devido à manutenção (*split job*), sendo a atividade anteriormente interrompida e reiniciando quando a máquina estiver disponível.

Em muitos processos industriais que operam em batelada, o tempo para preparar uma máquina às vezes é maior que o tempo de processamento, portanto, não se pode desprezar o tempo de *setup*. A programação de produção em ambiente *flowshop* considerando tempos de *setup* é bastante comum em plantas químicas, como por exemplo, indústria de tintas, de polpas de frutas, de tingimento de tecidos, de tintas e de detergente. Nestas indústrias, o tempo de preparação das máquinas é considerável e deve ser considerado na programação (SIMONS JR., 1992).

No que se refere ao processamento em batelada, onde, em regra, diversos produtos são manufaturados, pode-se dizer que dois tipos de tempos de *setup* de máquinas podem ser considerados. Cita-se a primeira situação em que o tempo de *setup* depende somente da tarefa a ser executada (independente da sequência). Na segunda situação, o tempo de *setup* depende tanto da tarefa a ser executada quanto daquela que foi processada imediatamente antes, portanto dependente da sequência.

Quando o sequenciamento de tarefas envolve única máquina e quando são considerados tempos de *setup*, o problema de minimização do tempo de processamento equivale a encontrar a sequência de tarefas que leva ao mínimo total *setup times*, uma vez que o somatório dos tempos de processamento das tarefas é constante. Assim, o problema passa a ser semelhante ao clássico problema do caixeiro viajante (BAKER, 1974). Em regra, a modelagem matemática é utilizada para resolver problemas de sequenciamento com tempos de *setup* (ALI ALLAHVERDI *et al*., 2008; LEE *et al*., 2011), sendo utilizadas técnicas de programação linear inteira-mista (SRIKAR e GHOSH, 1986; STAFFORD e TSENG, 1990; RIOS-MERCADO e BARD, 1998). Além da modelagem matemática, métodos heurísticos (SOLIMANPUR e ELMI, 2013; WANG, 2013) e métodos de enumeração implícita, tal como *branch and bound* também podem ser utilizados.

O método *branch and bound* utiliza-se busca controlada guiada por uma função objetivo (*lower bound*), a qual deve ser estabelecida adequadamente para cada problema. Nilson (1971) aborda alguns métodos de redução de espaço, dentre eles a pesquisa *depth first*, onde os nós com

maior nível de profundidade na árvore são escolhidos, e a pesquisa *best first*, onde os nós mais promissores são escolhidos para a ramificação.

O procedimento de cálculo de limitante é crucial para que a estratégia de enumeração implícita forneça solução ótima do problema proposto. Para a redução do espaço de busca é necessário que esse limitante da função objetivo seja estimado com garantia de não exclusão de regiões factíveis.

3. Algoritmo de busca em árvore branch and bound

3.1. Relações de recorrência sem considerar tempo de setup para o sequenciamento de N tarefas em M máquinas (*heurística full machine*)

Quando se parte para a solução de problemas de programação de produção via formulação matemática, duas restrições são básicas para a modelagem: a restrição de ordenamento das perações de diferentes tarefas em cada máquina e a restrição de precedência tecnológica, determinando assim que uma operação de uma determinada tarefa só pode ser desenvolvida depois do término da operação anterior (SANTOS, 1994).

Rodrigues e Latre (1992) desenvolveram um algoritmo, de sequenciamento e alocação de operações em sistemas flexíveis com restrições sobre recursos compartilhados e para o cálculo do *lower bound*, baseado na heurística *full machine*, sendo utilizadas as expressões de 1 a 5. Santos e Pereira (1974) estabeleceram algoritmo para o sequenciamento em ambiente *flowshop* envolvendo tempos de *setup* e ordem de prioridade na execução de tarefas.

a) Relação para o tempo de término de uma sequência de parcial da máquina j, dada pela Equação 1.

$$C_{(qi,j)} = \max\left\{C_{(qi,j-1)}; C_{(q,j)}\right\} + t_{ij} \qquad C_{i0} = C_{0j} = 0 \qquad (1)$$

onde t_{ij} é o tempo de processamento da tarefa i na máquina j e $C_{qi,j}$ o tempo de término de processamento da sequência parcial qi na máquina j.

b) Tempo de processamento restante, R_j, das tarefas ainda não processadas na máquina j, fornecida pela Equação .

$$R_j = \sum_{i \in U} t_{ij} \qquad \forall\, j \in M \tag{2}$$

onde U é o conjunto de tarefas não processadas e M o número de máquinas.

c) Tempo mínimo de trabalho, U_j, supondo que o processamento na máquina j já foi encerrado (Equação 3).

$$U_j = \min_{i \in U} \left\{ \sum_{k=j+1}^{M} t_{ik} \right\} \qquad \forall\, j \in M \tag{3}$$

d) *Lower bound*, b_j, na máquina j é dado pela Equação 4:

$$b_j = C(qi, j) + R_j + U_j \qquad \forall\, j \in M \tag{4}$$

sendo que o *lower bound* da função objetivo que orienta a busca é dado pela Equação 5:

$$LB = \max_{1 \le J \le M} \{ b_j \} \tag{5}$$

3.2. Método redutor da matriz de tempos de setup

A heurística *full machine* apresentada na seção 3.1 foi modificada para incorporação da matriz **S** de tempos de *setup* com elementos S_{nm}, n e m variando de 1 até N.

O método redutor proposto aplica-se em linhas e/ou colunas da matriz de tempos de *setup*, sendo a aplicação feita apenas em linha e/ou coluna que não tenham elementos nulos. No processo redução, os elementos de cada linha são diminuídos do menor elemento da linha. Caso a operação de redução em cada linha não produza pelo menos um zero em cada coluna, faz-se a redução de cada coluna.

Este procedimento é feito até que a sequência de tarefas seja encontrada. O método redutor segue as etapas:

a) Cálculo do custo de redução da enésima linha da matriz S de tempos de *setup*, θ_n,é realizado pela Equação 6.

$$\theta_n = \min_{m=1,2..N} \{ S_{nm} \} \qquad n = 1,2...N \tag{6}$$

em N é o número de tarefas.

b) Cálculo do custo de redução da coluna m é realizado pela Equação 7.

$$\theta_m = \min_{n=1,2,...N} \{S_{nm}\} \qquad m = 1,2,...N \tag{7}$$

c) O limitante inferior da redução r, θ_r, que é o custo total para surgir um zero em cada linha e coluna da matriz S de tempos de *setup* na etapa r, é dado pela Equação 8.

$$\theta_r = \sum_{i}^{N} \theta_n + \sum_{m}^{N} \theta_m \tag{8}$$

d) Rotular cada elemento zero com a soma dos valores mínimos da linha e coluna.

e) Formação dos subproblemas P_{ik}. Selecionar para fazer parte da sequência de tarefas da matriz de *setup*, após a redução, o subproblema P_{ik} formado pela linha i (tarefa i) e coluna k (tarefa k) do elemento da matriz que seja zero que possua rótulo com maior valor.

f) Eliminar pares de tarefas infactíveis. Como o subproblema P_{ik} faz parte da sequência não é mais possível acontecer o subproblema P_{ki}.

g) Promover nova redução após a seleção do P_{ik} e eliminação de problemas infactíveis caso a matriz não possua pelo menos um zero em cada linha e em cada coluna.

h) Enquanto houver pares de tarefas não inseridas na sequência de produção, repetir as etapas anteriores.

i) O limitante inferior total (θ_T), fornecido pela Equação 9, envolvendo tempos de *setup* e considerando todas as reduções r é dado pela Equação 9:

$$\theta_T = \sum_{r} \theta_r \tag{9}$$

3.3. Relações de recorrência propostas para o sequenciamento de tarefas com tempos de setup

O problema considerado neste trabalho é constituído de N tarefas a serem desenvolvidas em M máquinas, adotando o processamento *flowshop*. O tempo de término de uma sequência produção parcial em uma máquina é dado pela Equação 10:

$$C_{(qk,j)} = \max\{C_{(qk,j-1)}, C_{(q,j)} + s_{k',k,j}\} + t_{kj} \qquad C_{(qk0)} = C_{(0j)} = 0 \tag{10}$$

onde k é a tarefa inserida na sequência parcial q; k' é a tarefa que antecede à tarefa k na sequência parcial q; t_{kj} é o tempo de processamento da tarefa k na máquina j; $C_{qk,j}$ é o tempo de término de processamento da sequência parcial qk na máquina j; $s_{k',k,j}$ é o tempo de *setup* para preparar a máquina j para a tarefa k depois que a tarefa k' é executada.

Considerando a sequência parcial qk, Equação 11, o tempo mínimo para que se desenvolvam as atividades restantes na máquina j é dado por:

$$b_j^* = C_{(qk\,j)} + \left(\theta_T^{qk} - sc_{qk}^j\right) + R_j \qquad \forall\, j : \{1 \le j \le M\} \tag{11}$$

onde b_j^* é o *lower bound* da máquina j com tempo de setup; θ_T^{qk} é o *lower bound* da sequência de tarefas qk envolvendo somente os tempos de *setup* (calculado pela Equação 9). Considerando os pares de tarefas já presentes na sequência qk, sc_{qk}^j é o tempo de *setup* da sequência parcial qk na máquina j, sendo R_j, a soma dos tempos de processamento das tarefas ainda não executadas na máquina j, dado pela Equação 12.

$$R_j = \sum_{i \in U} t_{ij} \qquad \forall j \in M \tag{12}$$

Em que U é o conjunto de tarefas não processadas.

O limitante inferior para a sequência parcial qk é dado pela Equação 13

$$LB_{qk}^{**} = \max\left\{b_j^*\right\} \tag{13}$$

A busca em árvore *branch and bound* desenvolvida, neste trabalho, inicia-se fixando uma tarefa i, formando uma sequência parcial q. Posteriormente forma-se o subproblema P_{ik} pela inserção da tarefa k pertencente ao conjunto U e calcula o *lower bound* LB_{qk}^{**} (Equação 14). Para o cálculo do *lower bound* da sequência q é proposta a Equação Equação 14:

$$LB_q^{**} = \min\left\{LB_{qk}^{**}\right\} \tag{14}$$

4. RESULTADOS E DISCUSSÃO

O algoritmo proposto foi aplicado ao problema de Programação *flowshop* permutacional com N tarefas e com duas máquinas {M1, M2}. Foi utilizado computador HP Pavilion, 4 GB RAM,

processador intel core i3-350 e velocidade 2.26 GHz. Na Tabela 1 são apresentados os tempos de processamento das tarefas.

Tabela 1. Tempos de processamento nas Máquinas

Tarefas	M1	M2
1	4	5
2	2	3
3	6	3
4	1	4

Os tempos de *setup* das máquinas M1 e M2 estão dispostos nas Tabelas 2 e 3 respectivamente.

Tabela 2. Tempos de setup da máquina 1

Tarefas	1	2	3	4
1	-	6	7	3
2	5	-	1	4
3	2	7	-	2
4	8	4	3	-

Tabela 3. Tempos de setup na máquina 2

Tarefas	1	2	3	4
1	-	3	12	8
2	5	-	4	2
3	4	9	-	5
4	2	7	5	-

As Tabelas 4 e 7 mostram a geração dos nós da árvore de busca *branch and bound*, para o exemplo proposto. Para cada tarefa, o algoritmo inicia uma busca branch and bound gerando os nós correspondents as sequências parciais do sequenciamento. A Tabela 4 mostra que quando o sequenciamento é iniciado pela tafefa 1, tem-se a formação de uma sequência completa, 12341, com tempo de execução (makespan) de 33 u.t (unidades de tempo). Nas tabelas 5 e 6, nenhuma sequência completa foi formada, pois os tempos das sequências parciais foram maiores que 33 u.t. da sequência completa anteriormente citada.

Tabela 4. Subproblemas das sequências iniciando com a tarefa 1

Sequência parcial q	Tempo de término $(C_{q,1},C_{q,2})$	Sequência parcial qk	$(C_{qk,1},C_{qk,2})$	(b_1^*, b_2^*)	LB_{qk}^{**}	LB_q^{**}
1-2	(12,15)	1-2-3	(19,22)	(30,33)	33	33
1-2	(12,15)	1-2-4	(17,21)	(28,33)	33	
1-3	(17,24)	1-3-2	(26,36)	(39,44)	44	44
1-3	(17,24)	1-3-4	(20,33)	(31,48)	48	
1-4	(8,21)	1-4-2	(14,31)	(23,42)	42	42
1-4	(8,21)	1-4-3	(17,29)	(31,46)	46	
1-2-3	(19,22	1-2-3-4	(22,31)	(30,33)	33	33
1-2-4	(17,21)	1-2-4-3	(26,29)	(28,33)	33	33
1-2-3-4-1					33	

A Figura 2 mostra a árvore de busca para as sequências que iniciam com a tarefa 1. Nota-se que o algoritmo ao encontrar a solução completa 1-2-3-4-1, com *makespan* de 33, e ao proceder o retrocesso (*backtraking*) na árvore de busca irá eliminar os nós 3 e 4, pois possuem limitante inferior maior que a solução parcial encontrada.

Figura 2. Branch and bound para a sequencia iniciando com a tarefa 1.

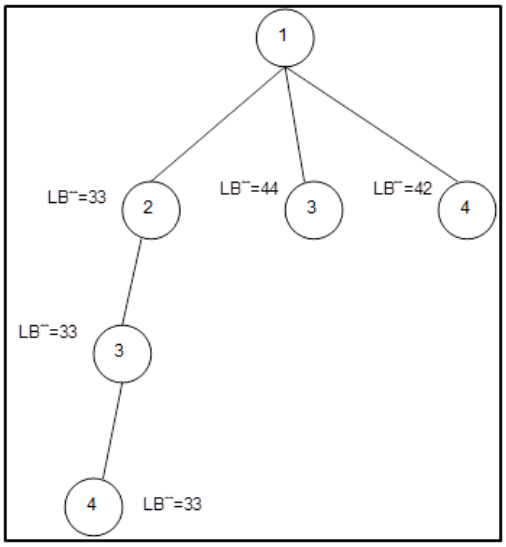


Tabela 5 - Subproblemas das sequências iniciando com a tarefa 2

Sequência parcial q	Tempo de término $(C_{q,1},C_{q,2})$	Sequência parcial qk	$(C_{qk,1},C_{qk,2})$	(b_1^*, b_2^*)	LB_{qk}^{**}	LB_q^{**}
2-1	(11,16)	2-1-3	(24,31)	(31,47)	47	45
2-1	(11,16)	2-1-4	(15,28)	(31,45)	45	
2-3	(9,12)	2-3-1	(15,21)	(23,40)	41	31
2-3	(9,12)	2-3-4	(12,21)	(30,31)	31	
2-4	(7,11)	2-4-1	(19,24)	(39,48)	48	31
2-4	(7,11)	2-4-3	(16,19)	(28,31)	31	
2-3-1	(15,21)	2-3-1-4	(19,33)	(23,40)	40	40
2-3-4	(12,21)	2-3-4-1	(24,29)	(30,32)	32	32

Tabela 6 - Subproblemas das sequências iniciando com a tarefa 3

Sequência parcial q	Tempo de término $(C_{q,1},C_{q,2})$	Sequência parcial qk	$(C_{qk,1},C_{qk,2})$	(b_1^*, b_2^*)	LB_{qk}^{**}	LB_q^{**}
3-1	(12,18)	3-1-2	(20,24)	(28,37)	35	35
3-1	(12,18)	3-1-4	(16,30)	(23,44)	44	
3-2	(15,21)	3-2-1	(24,31)	(31,48)	48	46
3-2	(15,21)	3-2-4	(20,27)	(39,46)	46	
3-4	(9,18)	3-4-1	(21,26)	(30,36)	36	36
3-4	(9,18)	3-4-2	(15,28)	(31,50)	50	

Tabela 7 – Subproblemas das sequências iniciando com a tarefa 4

Sequência parcial q	Tempo de término $(C_{q,1},C_{q,2})$	Sequência parcial qk	$(C_{qk,1},C_{qk,2})$	(b_1^*, b_2^*)	LB_{qk}^{**}	LB_q^{**}
4-1	(13,18)	4-1-2	(21,24)	(30,36)	36	36
4-1	(13,18)	4-1-3	(26,33)	(39,47)	47	
4-2	(7,15)	4-2-1	(16,25)	(31,45)	45	39
4-2	(7,15)	4-2-3	(14,22)	(23,29)	39	
4-3	(10,13)	4-3-1	(16,22)	(28,30)	30	30
4-3	(10,13)	4-3-2	(19,25)	(31,43)	43	
4-3-1	(16,22)	4-3-1-2	(24,28)	(28,30)	30	30
4-3-1-2-4*	(16,22)	4-3-1-2	(24,28)	(28,30)	30	

(*) sequência ótima

Tabela 8. Enumeração completa

Sequência	Makespan (u.t)
1-2-3-4-1	33
1-2-4-3-1	33
1-3-2-4-1	44
1-3-4-3-1	48
1-4-2-3-1	42
1-4-3-2-1	46
2-1-3-4-2	47
2-1-4-3-2	45
2-3-1-4-2	40
2-3-4-1-2	32
2-4-1-3-2	48
2-4-3-1-2	31
3-1-2-4-3	35
3-1-4-2-3	44
3-2-1-4-3	48
3-2-4-1-3	49
3-4-1-2-3	36
3-4-2-1-3	50
4-1-2-3-4	36
4-1-3-2-4	47
4-2-1-3-4	45
4-2-3-1-4	39
4-3-1-2-4*	30
4-3-2-1-4	43

(*) sequência ótima

O algoritmo encontrou a sequência completa 4-3-1-2-4 (Tabela 8) como melhor sequência e com *makespan* igual a 30 u.t (unidades de tempo). Este resultado pode ser comparado com a enumeração completa gerada a partir dos dados do exemplo proposto. O tempo computaciocal (CPU) gasto para este problema foi de 6.0 segundos. Foram testados problemas com 5 tarefas (tempo de CPU médio= 15 segundos) e 7 tarefas (tempo de CPU médio = 28 segundos). Mesmo abordando problemas distindos, pois na literatura da área não há registro do procedimento apresentado neste trabalho quando tempos de setup são considerados em sequenciamentode tarefas, Yamashita e Morabito (2007) mostram o uso do método branch and bound na solução de problemas de programação de produção envolvendo custo de disponilidade de recursos, no qual também avalia a eficiência da metodologia por tempos de CPU.

5. CONCLUSÃO

Neste artigo foi apresentado um algoritmo que se utiliza da busca *branch and bound* para o problema *flowshop* permutacional com tempos de preparação das máquinas dependentes da máquina. Foi desenvolvido um *lower bound* para resolver problemas de sequenciamento envolvendo tempos de *setup* com N tarefas e com M máquinas. Foi mostrada neste artigo a aplicação em um caso especial, que foi um *flowshop* com duas máquinas e 4 tarefas. Na solução de

problema somente foram geradas duas sequencia completas 1-2-3-4-1 e 4-3-1-2-4 (sequência ótima) de um total de 24 sequências possíveis, o que mostra a eficiência do algortimo.

REFERÊNCIAS

ALI ALLAHVERDI, C.T. NG B.; CHENG, T.C.E.; KOVALYOV, MIKHAIL Y. A survey of scheduling problems with setup times or costs. **European Journal of Operational Research**, v. 187, p. 985-1032, 2008.

BALAS, E. The prize collecting traveling salesman problem. **Networks,** v. 19, p. 621- 636, 1989

Baker, K. R., **Introduction to Sequencing and Scheduling**. New York: John Wiley, 1974.

CHAVES, A. A.; BIAJOLI, F. L.; MINE, O. M.; SOUZA, M. J. F. A. Metaheurísticas híbridas para resolução do problema do caixeiro viajante com coleta de prêmios. **Produção**, v. 17, n. 2, p. 263-272, 2007.

CHEN, W. J. Scheduling with dependent setups and maintenance in a textile company. **Computers & Industrial Engineering**, v. 57, p. 867-873, 2009.

GONÇALVES JÚNIOR; JESUS, E. Sequenciamento de produção com restrição na ordem de precêdencia**. GEINTEC**, v. 2, n. 3, p.274-284, 2012.

GOMES, L.; DINIZ, V.; MARTINHON, C. A. An Hybrid GRASP+VND Metaheuristic for the Prize-Collecting Traveling Salesman Problem. **In**: XXXII Simpósio Brasileiro de Pesquisa Operacional, p. 1657-1665, 2000.

HEJAZI, S.R.; SAGHAFIAN, S. Flowshop-scheduling problems with makespan criterion: a review. **International Journal of Production Research**, v. 43, p. 2895-2929, 2005.

KU, H.; RAJAGOPALAN, D.; KARIMI, I. Scheduling in Batch Process. **Chemical Engineering Progress**, v. 83, n. 8, p. 35-45, 1987.

LEE, Wen-Chiung.; LIN, Jian-Bang.; SHIAU, Yau-Ren. Deteriorating job scheduling to minimize the number of late jobs with setup times**. Computers & Industrial Engineering**, v. 61, p. 782-787, 2011.

MELO, V. A. **Metaheurísticas para o Problema do Caixeiro Viajante com Coleta de Prêmios**. 2001. Dissertação (Mestrado em Engenharia de Computação) – ProgramaUniversidade em Engenharia de Computação, Universidade Federal Fluminense, Niterói.

MOCCELLIN, J. V.; NAGANO, M. S. Uma propriedade estrutural do problema de programação da produção flow shop permutacional com tempos de setup. **Pesquisa Operacional**, v. 27, n. 3, p. 487-515, 2007.

NILSON, N. J. **Problem-solving methods in artificial intetelligence**. New York: McGraw-Hill Back Company. 1971.

REKLAITIS, G. V. *Review of scheduling of Processs operations*. AIche Symposium Series. p. 119,1982.

RIOS-MERCADO, R.Z.; BARD, J.F. Heuristics for the flow line problem with setup costs. **European Journal of Operational Research**, v. 110, p. 76-98, 1998.

RODRIGUES, M.T.M.; LATRE, G. **Sequenciamento e Alocação de Operações em Flow-Shop na Indústria Química com Restrições sobre os Recursos Compartilhados**: Uma Abordagem de Busca Orientada por Restrições. 1992. Tese (Doutorado em Engenharia Elétrica) – Programa de Pós-Graduação em Engenharia Elétrica, Universidade Estadual de Campinas, Campinas-SP.

SOLIMANPUR, M.; ELMI, A. A tabu search approach for cell scheduling problem with makespan criterion. **International Journal of Production Economics**, v. 141, n. 2, p. 639-645, 2013.

SANTOS, E. J. **Análise de Tempos de Estabelecimento e Ordem de Manufatura no Sequenciamento de Tarefas em Processos Batelada**.1994. Dissertação (Mestrado em Engenharia Química) - Programa de Pós-Graduação em Engenharia Química, Universidade Estadual de Campinas, Campinas-SP.

SANTOS, Edilson J. **Interferência Lógica em Problemas de Programação de Produção em Sistemas Flexíveis.** 1998. Tese (Doutorado em Engenharia Química) – Programa de Pós-Graduação em Engenharia Química, Universidade Estadual de Campinas, Campinas-SP.

SIMONS Jr., J.V. Heuristics in flow shop scheduling with sequence dependent setup times. **Omega**, 20, p. 215-225, 1996.

SRIKAR, B. N.; GHOSH, S. A. MILP model for the n-job, m-stage flowshop, with sequence dependent setup times. **International Journal of Production Research**, 24, p. 1459-1472, 1986.

STAFFORD, E.F.; TSENG, F.T. On the Srikar-Ghosh MILP model for the N X M SDST flowshop problem. **International Journal of Production Research**, 28, p. 1817-1830, 1990.

USKUP, E.; SMITH, S. B. A branch-and-bound algorithm for two-stage production-sequencing problem. **Operations Research**, 23, p. 118-136, 1975.

WANG, X.; XIE, X.; CHENG, T.C.E. Order acceptance and scheduling in a two-machine flowshop. **International Journal of Production Economics**, v. 141, n 1,p. 366-376, 2013.

YAMASHITA, D. S; MORABITO, R. Um algoritmo branch-and-boundpara o problema de programação de projetos com custo de disponibilidade de recursos e múltiplos modos. **Gest. Prod**., v. 14, n. 3, p. 545-555, 2007.

Recebido: 30/05/2014

Aprovado: 15/06/2014