DOI: 10.5585/ExactaEP.v11n2.4466

Estudo da influência da programação do primeiro estágio em *flow shop* híbridos com tempos de *setup* explícitos

Influence of first-stage schedule on hybrid flow shop with separated setup times

Helio Yochihiro Fuchigami
Doutor em Engenharia de Produção pela EESC/USP,
Departamento de Matemática/Matemática Industrial.
Universidade Federal de Goiás – UFG.
Catalão, GO [Brasil]
heliofuchigami@yahoo.com.br

João Vitor Moccellin
Doutor em Engenharia Mecânica pela EESC/USP,
Departamento de Engenharia Mecânica e de Produção,
Universidade Federal do Ceará/Campus do Pici – UFC.
Fortaleza, CE [Brasil]
jvmoccel@sc.usp.br

Resumo

Neste artigo são apresentados quatro métodos heurísticos construtivos (denominados LPT, TOTAL, SCT e LPST) para programação da produção em sistemas *flow shop* híbridos com tempos de *setup* independentes da sequência de execução das tarefas. O foco deste trabalho é o estudo da influência da programação do primeiro estágio nesse ambiente de produção. O critério de desempenho utilizado é a minimização da duração total da programação (*makespan*). O método LPT baseia-se na conhecida regra *Longest Processing Time*, que sequencia as tarefas pela ordem crescente da soma dos tempos de processamento de todos os estágios. O TOTAL utiliza a ideia do Método de Aproximação de Vogel, considerando a matriz composta pela soma dos tempos de processamento e *setup* de todos os estágios. A denominação SCT foi atribuída a este método por escolher o par tarefa-máquina que conduz à menor data de término (*Shortest Completion Time*). E o método LPST atribui à máquina de menor carga a tarefa com a maior soma dos tempos de processamento e *setup* do primeiro estágio. Os resultados da experimentação computacional mostraram que o método SCT foi claramente superior aos demais, com 55,3% de sucesso, enquanto o LPT obteve 26,9%, o TOTAL atingiu 16,4% e, por último, o LPST apresentou 2,2% de sucesso.

Palavras-chave: *Flow shop* híbrido. Programação da produção. *Setup* independente.

Abstract

This study addresses the problem of hybrid flow shop scheduling with sequence-independent setup times. Four constructive heuristic methods (called LPT, TOTAL, SCT and LPST) were proposed with the aim of assessing the impact of the schedule of the first stage of production in this environment. The minimization of the total time to complete the schedule (makespan) was used as a performance criterion. The LPT method is based on the well-known Longest Processing Time rule, which organize tasks in the ascending order of the sum of processing times of all the stages. The TOTAL method uses the idea of Vogel Approximation Method, considering the matrix composed by the sum of processing and setup times of all stages. The SCT method chooses the pair machine-job that leads the Shortest Completion Time. And the LPST method assigns to the lower load machine the job with the largest sum of processing and setup times of the first stage. The results of computing experiments showed that the SCT method was much higher, with 55.3% of success, while LPT obtained 26.9%, TOTAL reached 16.4% and lastly LPST presented 2.2% of success.

Key words: Hybrid flow shop. Production scheduling. Sequence-independent setup.

1 Introdução

Esta pesquisa aborda o problema de programação da produção em ambientes denominados *flow shop* híbridos, caracterizados por sistemas compostos por múltiplos estágios de produção, em que há, em cada um deles, mais de uma máquina operando em paralelo. Este problema também é conhecido como *flow shop* com máquinas múltiplas ou *flow shop* flexível e envolve a programação de um conjunto de *n* tarefas que estão disponíveis para serem processadas por um conjunto de *g* estágios de produção em fluxo linear unidirecional. Cada um dos estágios pode conter duas ou mais máquinas paralelas consideradas idênticas. E cada tarefa deve ser processada em exatamente uma máquina em cada estágio.

No estudo, foi considerada como medida de desempenho a duração total da programação (*makespan*) e os tempos de *setup* foram tratados de forma explícita, ou seja, separados dos tempos de processamento, conforme Allahverdi et al. (2008). Além disso, os tempos de *setup* são independentes da sequência de execução das tarefas.

O tempo de *setup* ou tempo de preparação das máquinas representa as operações realizadas antes da execução do produto ou do início do processamento da tarefa. Inclui todo o trabalho de preparação da máquina ou da oficina para a fabricação de produtos, por exemplo, a obtenção e ajuste de ferramentas, inspeção e posicionamento de materiais e processos de limpeza.

Em sua revisão de problemas com tempos e custos de *setup*, Allahverdi et al. (2008) afirmaram que muitas pesquisas em programação da produção desconsideram os tempos de *setup* ou os incluem nos tempos de processamento das tarefas. Entretanto, algumas situações exigem que o *setup* seja separado dos tempos de processamento. Em uma fábrica de embalagens de papel, por exemplo, a duração do *setup* entre as trocas dos diferentes tipos de embalagens depende do grau de similaridade (como tamanho e número de cores) entre os lotes consecutivos. Desta forma, o tempo de *setup* não pode ser incluído no tempo de processamento das tarefas (PINEDO, 2010). Outra aplicação com tempos de *setup* separados é a indústria de tinta em que o processo de limpeza da máquina depende da sequência de cores produzidas (CONWAY et al., 1967). Situações práticas semelhantes são encontradas nas indústrias química, farmacêutica, alimentícia, metalúrgica e de semicondutores (ALLAHVERDI, 2000). Esses casos evidenciam a importância prática desta pesquisa.

Existem dois tipos de problemas com o tempo de *setup* separado do tempo de processamento das tarefas. No primeiro caso, o *setup* depende somente da tarefa a ser processada e é chamado "independente da sequência"; e no segundo, depende tanto da tarefa a ser processada como também da que foi executada imediatamente antes na mesma máquina, sendo denominado "dependente da sequência" (ALLAHVERDI, GUPTA; ALDOWAISAN, 1999).

Neste trabalho, o conceito de *setup* foi tratado de forma genérica, com duração fixa e conhecida previamente, representando quaisquer atividades exemplificadas anteriormente (ajustes, posicionamentos, limpeza, entre outros).

Uma importante implicação dos tempos de *setup* separados dos tempos de processamento é que a operação de *setup* na máquina subsequente pode ser iniciada enquanto a tarefa ainda está sendo executada na máquina imediatamente precedente (ALLAHVERDI, 2000). Isso ocorre porque, como o *setup* e o processamento são programados separadamente, não é preciso esperar a conclusão do processamento em uma máquina para iniciar o *setup* na máquina seguinte.

Por exemplo, como afirma Allahverdi (2000), geralmente há tempo ocioso nas máquinas do segundo estágio, pois elas precisam esperar

o processamento das tarefas no primeiro estágio. Assim, enquanto a tarefa ainda está sendo processada no primeiro estágio, as operações de *setup* do segundo estágio podem ser realizadas, ou seja, o *setup* pode ser antecipado em relação à liberação da tarefa no estágio anterior. Isto significa que uma medida de desempenho regular pode ser melhorada considerando os tempos de *setup* separados dos tempos de processamento.

Objetivou-se, neste trabalho, analisar como a programação do primeiro estágio de produção afeta a duração total da programação (*makespan*) em ambientes *flow shop* híbridos com tempos de *setup* independentes da sequência de processamento das tarefas.

2 Programação da produção em *flow shop* híbridos

Os ambientes industriais complexos, com a presença de várias máquinas em paralelo nos diversos estágios de produção, são referenciados como *flow shop* híbridos, *flow shop* com múltiplas máquinas, *flow shop* com múltiplos processadores ou *flexible flow shop*, conforme descritos a seguir.

Vignier, Billaut e Proust (1999) apresentaram o estado da arte para aquele momento para *flow hops* híbridos. Linn e Zhang (1999) propuseram uma classificação das pesquisas conforme o número de estágios de produção do problema.

Variações flexíveis de *flow shops* híbridos podem ser encontradas em Vairaktarakis (2004). Kis e Pesch (2005) atualizaram o estado da arte com trabalhos posteriores a 1999, enfocando métodos de solução exata para minimização do *makespan* e tempo médio de fluxo. Wang (2005) fez uma revisão da literatura, classificando em métodos de solução ótima, heurística e de inteligência artificial. Quadt e Khun (2007) publicaram uma taxonomia para *flow shop* híbrido, porém denominando-o de *flexible flow line*.

Ruiz e Vázquez-Rodríguez (2010) apresentaram uma revisão da literatura de métodos exatos, heurísticos e meta-heurísticos, discutindo as variações do problema, suas diferentes hipóteses, restrições, funções e objetivos, além de apresentar as oportunidades de pesquisa na área. E uma extensa revisão dos trabalhos publicados recentemente (desde 1995) foi elaborada por Ribas, Leisten e Framiñan (2010), com um novo método de classificação dos trabalhos, do ponto de vista da produção, de acordo com as características das máquinas e das tarefas.

Os sistemas *flexible flow shop* estão se tornando gradativamente mais frequentes na indústria, principalmente devido à grande carga de trabalho requerida pelas tarefas nas máquinas (LOGENDRAN; CARSON; HANSON, 2005). Como observaram Quadt e Kuhn (2007), o sistema *flow shop* híbrido pode ser encontrado em um vasto número de indústrias, tais como químicas, eletrônicas, de empacotamentos, farmacêuticas, automotivas, de fabricação de embalagens de vidro, madeireiras, têxteis, de herbicidas, alimentícias, de cosméticos e de semicondutores.

Entretanto, embora tenham sido feitos muitos trabalhos nesta área de programação da produção, conforme os apresentados nesta seção, a maioria se restringiu a casos especiais de dois estágios ou de configurações específicas de máquinas nos estágios, ou então sem a presença de tempos de *setup* separados dos tempos de processamento.

Mais recentemente, várias estratégias de alocação foram apresentadas por Weng, Wei e Fujimura (2012) visando a auxiliar regras de prioridade na programação de *flow shop* híbridos com chegadas dinâmicas de tarefas e filosofia *just-in-time* (JIT), isto é, objetivando reduzir adiantamentos e atrasos das tarefas.

Um estudo de caso em uma indústria de células de painéis solares, identificada como um *flow shop* híbrido, foi desenvolvido por Chen et al. (2013), considerando tempos de *setup* explícitos, havendo tanto dependentes como independentes da sequência. O objetivo nesse estudo foi a minimização do *makespan*.

Especificamente considerando os tempos de *setup* independentes da sequência de execução das tarefas, foram encontrados os trabalhos de Gupta e Tunc (1994), Li (1997) e Huang e Li (1998), nos quais se considerou o ambiente *flow shop* híbrido com dois estágios, sendo uma única máquina no primeiro e várias máquinas paralelas idênticas no segundo, com o critério de minimização do *makespan*.

Ainda com *setup* independente, Botta-Genoulaz (2000) e Allaoui e Artiba (2004) estudaram medidas que incluem o atraso máximo das tarefas. Low (2005) enfocou a minimização do tempo total de fluxo; e Logendran et al. (2005) propuseram heurísticas construtivas para otimizar o *makespan*.

Hekmatfar et al. (2011) estudaram um *flow shop* híbrido com dois estágios e minimização do *makespan* com possibilidade de reentrada de tarefas. Foram propostas heurísticas e meta-heurísticas para resolver o problema.

Um método de solução composto por um algoritmo genético e busca em vizinhança variável foi apresentado por Behnamian e Fatemi Ghomi (2011) para programação em *flow shop* híbrido com tempos de *setup* dependentes da sequência e tempos de processamento dependentes da velocidade da máquina e da quantidade de recursos a ela alocada no momento da execução da tarefa.

Gómez-Gasquet et al. (2012) desenvolveram um algoritmo genético melhorado para minimização do *makespan* em *flow shop* híbrido com *setup* dependente, cujo desempenho relativo a métodos publicados anteriormente foi comprovado.

Um método de solução exata *branch-and-bound* foi apresentado por Fattahi et al. (2013) para o *flow shop* híbrido com operações de montagem. O diferencial desse problema é a existência de um estágio de montagem ao final da produção.

Como pode ser observado, embora o ambiente *flow shop* híbrido tenha sido extensivamente estudado, principalmente sem o tratamento explícito do *setup*, ou então considerando *setup* dependente da sequência, existem relativamente poucas publicações abordando especificamente sistemas com *setup* independente, conforme tratado neste estudo. Isto salienta o potencial de pesquisa neste tipo de problema e a importância de propor novos métodos de solução, verificando inclusive suas propriedades estruturais, como a influência da programação do primeiro estágio de produção.

3 Métodos heurísticos construtivos propostos

O ambiente de produção *flow shop* híbrido é ilustrado na Figura 1. O problema consiste em programar um conjunto de n tarefas, definido como $J = \{J_1, ..., J_n\}$, em que cada tarefa possui necessariamente uma única operação em cada estágio de produção. As operações das tarefas devem ser realizadas sequencialmente e passam por todos os estágios. A medida de desempenho do problema é a minimização do *makespan*.

Para obter a solução deste problema de programação da produção, foram desenvolvidos e avaliados quatro métodos heurísticos construtivos com base em algoritmos reportados na literatura para solução de problemas em ambientes cujos tempos de *setup* são considerados separados dos tempos de processamento.

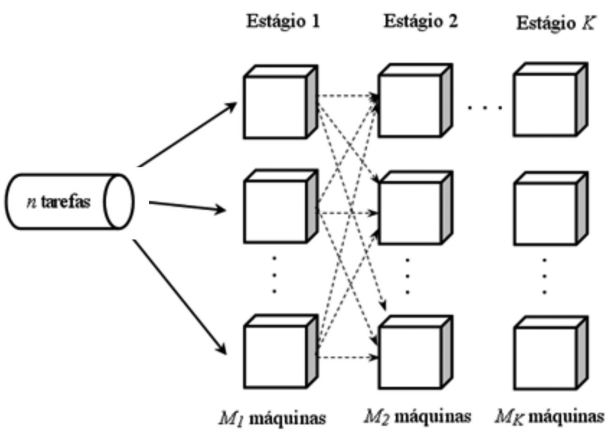

Figura 1: Ilustração do ambiente de produção

Foram definidos procedimentos para programação das tarefas estágio a estágio, como solução iterativa de K problemas relacionados. O primeiro estágio (k = 1) é programado como se fosse um problema tradicional de M_1 máquinas paralelas idênticas com a mesma data de liberação das tarefas, convencionada igual a zero ($r_i = r = 0$). Os estágios seguintes ($k \geq 2$) consistem em K–1 problemas consecutivos de M_k máquinas paralelas idênticas com diferentes datas de liberação das tarefas, correspondentes às datas de término das operações das respectivas tarefas no estágio imediatamente anterior (k–1).

Foram definidos quatro métodos de solução para este problema, denominados *Longest Processing Time* (LPT), TOTAL, *Shortest Completion Time* (SCT) e *Longest Processing-Setup Time* (LPST). Estes procedimentos diferem apenas na programação do primeiro estágio, uma vez objetiva-se neste trabalho verificar a variação do *makespan* dependendo da política de alocação utilizada neste estágio de produção.

Os procedimentos de programação para o primeiro estágio são detalhados a seguir. Em todos os algoritmos, considera-se que todas as máquinas estejam preparadas ao receber a primeira tarefa.

- Método LPT: define uma ordenação inicial para as tarefas de acordo com a regra de prioridade denominada *Longest Processing Time* (LPT), que considera a soma dos tempos de processamento de cada tarefa em todos os estágios e faz o sequenciamento pela ordem não decrescente dessas somas.
- Método TOTAL: estabelece uma ordenação inicial de tarefas com base na heurística TOTAL de Simons Jr. (1992), que utiliza o Método de Aproximação de Vogel (MAV), conhecido por fornecer boas soluções iniciais para problemas de transporte. O MAV seleciona células de uma matriz de transporte examinando o aumento da diferença entre os dois menores valores de cada linha e coluna. A célula selecionada em cada iteração é aquela que produz a maior diferença em comparação com a célula de menor valor na linha ou coluna. Neste trabalho, a matriz contém a soma dos tempos de processamento e de *setup* de todos os estágios de produção. Cada célula representa um possível par consecutivo de tarefas, e cada iteração seleciona uma sequência parcial de duas tarefas.

Os métodos LPT e TOTAL, que utilizam regras de ordenação inicial, alocam sequencialmente as tarefas priorizando a máquina cuja programação possuirá a data mais cedo de término.

- Método SCT: emprega uma adaptação do melhor entre os sete algoritmos para máquinas paralelas com *setup* dependente apresentados por Weng, Lu e Ren (2001), analisando todas as possibilidades de combinação tarefa-máquina e escolhendo o par com data de término mais cedo (*Shortest Completion Time*).
- Método LPST: atribui à máquina de menor carga a tarefa com maior valor da soma dos

tempos de *setup* e de processamento (*Longest Processing-Setup Time*) do primeiro estágio.

O procedimento para programação nos estágios subsequentes ao primeiro ($k \geq 2$) respeita a ordenação pela regra *Earliest Release Date* (ERD), que considera as datas de término das tarefas no estágio anterior como as datas de liberação do estágio atual. Isto significa que as tarefas serão programadas na ordem em que foram concluídas no estágio anterior, associando-as à máquina cuja programação possuirá a data mais cedo de término incluindo a tarefa analisada.

Como o foco do estudo aqui apresentado é a programação do primeiro estágio, os estágios $k \geq 2$ utilizam o mesmo procedimento de programação.

4 Experimentação computacional

4.1 Método da pesquisa

Segundo as definições de Martins (2010, p. 45-49) e Nakano (2010, p.64), esta pesquisa possui abordagem "quantitativa", pois há preocupação com mensurabilidade, causalidade, generalização e replicação. Pode também ser classificada como "experimento", uma vez que testa o relacionamento entre as variáveis da pesquisa operacionalizada, manipulando variáveis independentes (neste caso, a programação do problema) para se observar o resultado na variável dependente (representada aqui pela medida de desempenho *makespan*).

Além disso, de acordo com Jung (2004), este trabalho se classifica como "pesquisa aplicada" quanto à natureza, por gerar conhecimento com finalidades de aplicação prática, "pesquisa exploratória" quanto aos objetivos, pois visa à melhoria teórico-prática de sistemas, processos e produtos, e inovação pela proposição de novos modelos, além de ser feita a partir de impulsos criativos, simulações e experimentações, podendo originar novos modelos destinados a invenções, inovações e a otimização, e "pesquisa experimental" quanto aos procedimentos para a obtenção de novos conhecimentos e produtos tecnológicos, requerendo uma manipulação de variáveis detalhada e sistemática, e originando inovações a partir de ensaios e estudos dinâmicos em laboratório.

4.2 Planejamento do experimento

Na experimentação computacional, optou-se por gerar os problemas-teste a serem resolvidos de acordo com parâmetros considerados em trabalhos publicados na literatura específica. Os parâmetros utilizados foram: número de tarefas (n), número de estágios de produção (K) e por um conjunto de seis relações entre as ordens de grandeza dos tempos de processamento e de *setup* ($O(p_j)/O(s_{ij})$).

Foram gerados problemas-teste para cada opção do número de tarefas definida no conjunto {10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120}, considerando duas possibilidades do número de estágios {4, 7}, e uma das seis relações de grandeza dos tempos de processamento de *setup*, cuja definição será detalhada mais adiante.

Assim, as 12 opções do número de tarefas, as 2 opções do número de estágios e as 6 relações dos tempos de processamento e de *setup* perfazem 144 classes de problemas ($12 \times 2 \times 6 = 144$). Para cada classe de problemas, foram gerados cem problemas-teste visando a reduzir o erro amostral, totalizando 14.400 problemas-teste analisados.

Em cada estágio, o número de máquinas paralelas idênticas varia de dois a cinco, ou seja, $M_k \in \{2, 3, 4, 5\}$. Como M_k é gerado aleatoriamente com distribuição uniforme dentro deste conjunto, sua variação não altera a quantidade de problemas-teste gerados.

O conjunto das seis relações $O(p_j)/O(s_{ij})$, apresentado na Tabela 1, foi definido com base em trabalhos reportados na literatura de problemas com tempos de *setup* separados dos tempos de processamento. A geração dos tempos dentro de cada intervalo foi feita de forma aleatória com distribuição uniforme.

Tabela 1: Relações entre as ordens de grandeza dos tempos de processamento e de setup

Relação	Notação	Intervalo p_j	Intervalo s_{ij}
Relação I	$O(p_j)/O(s_{ij}) = 1$	1-99	1-99
Relação II	$O(p_j)/O(s_{ij}) < 1$	1-99	100-120
Relação III	$O(p_j)/O(s_{ij}) > 1$	10-99	1-9
Relação IV	$O(p_j)/O(s_{ij}) > 1$	50-99	1-49
Relação V	$O(p_j)/O(s_{ij}) \leq 1$	1-99	1-120
Relação VI	$O(p_j)/O(s_{ij}) \geq 1$	1-99	1-20

Simons Jr. (1992), Rajendran e Ziegler (1997) e Ríos-Mercado e Bard (1998) utilizaram intervalos comuns de tempos de processamento e de *setup*, com valores de 1 a 99, definindo os valores da relação I, denotada por $O(p_j)/O(s_{ij}) = 1$.

As relações II, III e IV são variações de ordens não comuns. De forma semelhante a Das, Gupta e Khumawala (1995) e Li (1997), valores dos tempos de processamento menores (1 a 99) do que os tempos de *setup* (100 a 120) são utilizados na relação II ($O(p_j)/O(s_{ij}) < 1$). Com base no trabalho de Ríos-Mercado e Bard (1999), valores dos tempos de processamento maiores (10 a 99) do que os tempos de *setup* (1 a 9) foram definidos na relação III e intervalos de tempos de processamento (50 a 99) e de *setup* (1 a 49) com amplitudes próximas são considerados na relação IV. Tanto a relação III como a IV são representadas por $O(p_j)/O(s_{ij}) > 1$.

Intervalos comuns de tempos de processamento e de *setup*, mas com amplitudes diferentes, foram estabelecidos nas relações V e VI. Intervalos de tempos de processamento menores (1 a 99) do que os de *setup* (1 a 120) são considerados na relação V ($O(p_j)/O(s_{ij}) \leq 1$). E, como definido por Gupta e Tunc (1994) e Weng, Lu e Ren (2001), intervalos de tempos de processamento maiores (1 a 99) do que os tempos de *setup* (1 a 20) foram utilizados na relação VI ($O(p_j)/O(s_{ij}) \geq 1$).

De acordo com esses parâmetros, todos os problemas foram gerados aleatoriamente por meio de um programa construído especificamente para esta finalidade. Para resolver os problemas foi desenvolvido o *software* que tanto pode solucionar os problemas de uma classe, utilizando cada um dos quatro métodos separadamente, quanto resolver os quatro métodos simultaneamente, gerando arquivos comparativos com o *makespan* e o tempo de computação.

Foi utilizado o sistema operacional Windows e a linguagem de programação Delphi. As configurações do computador em que se realizaram os experimentos são as seguintes: processador Pentium 4 da Intel, com 2 GHz de frequência.

4.3 Análise dos resultados

Os resultados obtidos na experimentação computacional foram analisados por meio da porcentagem de sucesso, desvio relativo, desvio-padrão do desvio relativo e tempo médio de computação dos quatro métodos desenvolvidos para cada classe de problemas.

A porcentagem de sucesso é calculada pelo número de vezes que o método forneceu a melhor solução (empatando ou não) dividido pelo número de problemas da classe.

O desvio relativo (DR_h) mede a variação correspondente à melhor solução obtida pelos métodos e é calculado da seguinte forma:

$$DR_h = \frac{D_h - D^*}{D^*} \quad (1)$$

em que D_h é o *makespan* obtido pelo método h, e D^* é o melhor *makespan* obtido pelos quatro métodos.

O desvio-padrão de uma amostra mede o grau de dispersão dos elementos em torno da média. Neste trabalho, o desvio-padrão do desvio relativo é o valor da variação dos desvios relativos de uma classe de problemas em torno do desvio relativo médio. Quanto menor for o valor do desvio-padrão, melhor é o método de solução, quando comparado com outro, no caso em que ambos apresentarem desvios relativos médios com diferença não significativa. Além disso, quanto menor o desvio-padrão, mais estável é o método no que se referente ao seu desempenho.

O desvio-padrão (S_h) do desvio relativo de um método (h) é calculado como segue:

$$S_h = \sqrt{\frac{\sum_{i=1}^{L} (DR_{h_i} - DRM)^2}{L-1}} \quad (2)$$

em que L é o número de problemas da classe, DR_{h_i} é o desvio relativo da solução do problema i, e DRM é o desvio relativo médio da classe de problemas.

O tempo médio de computação foi medido em milissegundos (ms).

A comparação da porcentagem de sucesso entre os quatro métodos de solução, agregando as seis relações $O(p_j)/O(s_{ij})$, é mostrada no gráfico da Figura 2.

Na totalidade dos problemas resolvidos, o desempenho do método SCT mostrou-se claramente superior aos demais, apresentando a melhor solução 7.962 vezes num total de 14.400 problemas, correspondendo a 55,3% de sucesso, conforme pode ser observado na Tabela 2, que apresenta os resultados em ordem de classificação dos métodos. Em seguida, o LPT obteve 26,9% de sucesso, o TOTAL atingiu 16,4% e, por último, o LPST apresentou 2,2% de sucesso.

Uma verificação relevante é que a totalização da Tabela 2 ultrapassa a quantidade de 14.400 problemas-teste analisados, e também soma das porcentagens fica acima dos 100%, por se in-

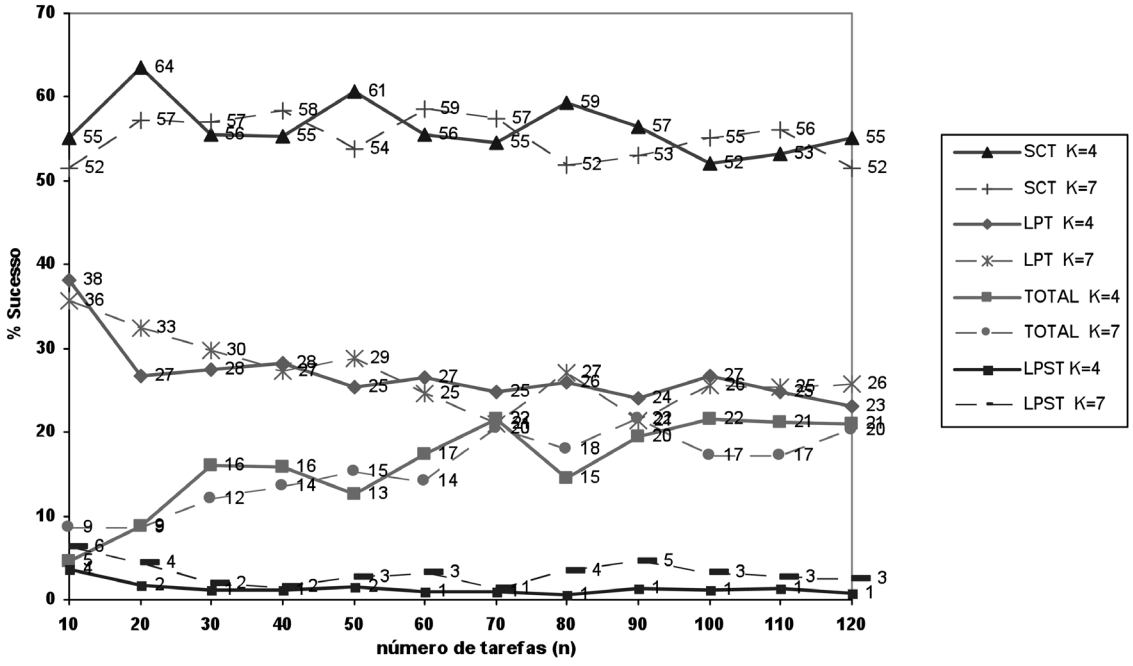

Figura 2: Comparação da porcentagem de sucesso entre os métodos agregando as relações $O(p_j)/O(s_{ij})$

Tabela 2: Total geral das porcentagens de sucesso

	SCT	LPT	TOTAL	LPST
Quantidade de problemas	7962	3875	2356	314
% total média	55,3	26,9	16,4	2,2

cluir os empates na porcentagem de sucesso dos métodos. Por exemplo, quando dois métodos forneceram o mesmo resultado, sendo este o melhor dentre os demais, ambos foram vitoriosos e é contabilizado o percentual de sucesso para ambos.

Em termos da variação do desempenho de cada método com o aumento do número de tarefas, ou seja, do porte do problema, o método SCT mantém seus resultados numa mesma faixa (em média de 52% a 64% de sucesso) com certa variação na amplitude. O método LPST mantém relativa estabilidade nos resultados (em torno de 2% de sucesso). O desempenho do método TOTAL melhora com o aumento do número de tarefas, principalmente nas relações I, II e V. E o método LPT reduz levemente o seu desempenho ao ampliar o porte do problema.

Como também podem ser observados no gráfico da Figura 2, os resultados dos quatro métodos para ambientes com quatro e sete estágios foram muito próximos, podendo indicar que o número de estágios não afeta o desempenho de um método e que outras opções do número de estágios teriam resultados semelhantes.

Na análise detalhada para cada relação $O(p_j)/O(s_{ij})$, existe certa alternância do melhor desempenho entre os métodos SCT e TOTAL, como pode ser visto na Tabela 3. O método SCT, que ocupou o primeiro lugar no resultado global, manteve também sua superioridade nas relações II, III, IV e VI. Porém, nas relações I e V, o método TOTAL obteve o melhor desempenho. Isto evidencia a importância da análise detalhada por parâmetros do problema, pois a média dos resultados pode ocultar diferenças no exame global.

Tabela 3: Porcentagens de sucesso dos métodos para cada relação $O(p_j)/O(s_{ij})$

Relações	SCT	LPT	TOTAL	LPST
Relação I	42,0	13,6	43,0	1,5
Relação II	51,9	18,0	29,2	1,7
Relação III	61,3	27,1	11,3	1,2
Relação IV	47,8	15,6	34,8	2,4
Relação V	36,1	12,8	50,0	1,5
Relação VI	61,5	22,5	15,0	1,2

Os valores dos desvios relativos e dos desvios-padrão do DR confirmam a análise feita para a porcentagem de sucesso dos métodos, inclusive para a ordem de superioridade destes, conforme apresenta a Tabela 4. O SCT, com o melhor desempenho na porcentagem de sucesso, teve os menores desvios, enquanto os valores do LPST foram os piores (exceto no tempo de computação).

Tabela 4: Desvios relativos, desvios-padrão e tempo de computação dos métodos

	SCT	LPT	TOTAL	LPST
Desvio relativo	2,0	5,2	5,1	18,7
Desvio-padrão do DR	0,03	0,05	0,07	0,14
Tempo de computação (ms)	5,1	4,5	17,3	4,9

A análise detalhada revelou que os métodos LPT, SCT e LPST mostraram uma redução nos valores do desvio-padrão do DR, com o aumento do porte do problema, mantendo certa estabilidade para problemas de grande porte. O método TOTAL apresentou uma considerável instabilidade em todas as relações $O(p_j)/O(s_{ij})$.

Como esperado, em todos os casos, o tempo de computação se eleva com o aumento do porte do problema. O método TOTAL tem o crescimento mais acentuado, o que pode ser explicado pela manipulação de matrizes, utilizada no Método de Simons Jr. (1992) e implementada no código computacional.

O tempo de computação, medido em milissegundos, não foi significativo, pois a média de

execução dos problemas foi a de 4,4 ms, e o maior tempo CPU atingiu apenas 35,5 ms.

5 Considerações finais

Nesta pesquisa, foi comprovada a influência da programação do primeiro estágio em ambientes *flow shop* híbridos com tempos de *setup* independentes da sequência de processamento das tarefas. A verificação foi revelada pelos valores do *makespan* consideravelmente diferentes, bem como o desempenho em termos de porcentagens de sucesso, quando se alterou, por meio de quatro regras distintas, o método de alocação e sequenciamento no primeiro estágio de produção.

O método mais flexível, SCT, que analisa um maior número de opções ao selecionar a tarefa e a máquina a que será designada, forneceu o melhor resultado. Já métodos baseados puramente em critérios de prioridade, como o LPT e o LPST, e o método TOTAL, que utiliza matrizes holísticas (composta com dados de todos os estágios), obtiveram resultados piores.

Estes resultados podem contribuir para melhorias nos processos produtivos em ambientes industriais compatíveis com o problema estudado. Podem ainda servir como ponto de partida para novas pesquisas e análises de problemas correlacionados.

A importância teórica deste estudo reside na mensuração da amplitude da diferença dos resultados de distintos métodos de solução. Optou-se por criar métodos de programação apenas para o primeiro estágio, mantendo-se o mesmo critério de alocação nos demais estágios, para avaliar de forma precisa o impacto da ordenação inicial. Isto sinaliza um estudo futuro da análise da influência dos demais estágios de produção, principalmente na identificação do estágio gargalo de produção (etapa restritiva no processo, geralmente com menor capacidade ou com maior demanda de processamento).

Sugere-se ainda para o desenvolvimento de futuros trabalhos a incorporação de novas restrições no problema, como, por exemplo, diferentes datas de liberação das tarefas e/ou das máquinas, e a utilização de outras medidas de desempenho, como a minimização do tempo médio de fluxo e dos adiantamentos e atrasos. Além disso, novas regras de ordenação inicial e de alocação para o primeiro estágio de produção também poderão ser testadas, visando a obter eventualmente melhor desempenho.

Agradecimentos

Os autores agradecem o apoio financeiro da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) na realização desta pesquisa.

Referências

ALLAHVERDI, A. Minimizing mean flowtime in a two-machine flowshop with sequence-independent setup times. *Computers & Operations Research*, v. 27, p. 111-127, 2000.

ALLAHVERDI, A.; GUPTA, J. N. D.; ALDOWAISAN, T. A review of scheduling research involving setup considerations. *Omega – The International Journal of Management Science*, v. 27, p. 219-239, 1999.

ALLAHVERDI, A. et al. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, v. 187, p. 985-1032, 2008.

ALLAOUI, H.; ARTIBA, A. Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints. *Computers and Industrial Engineering*, v. 47, p. 431-450, 2004.

BEHNAMIAN, J.; FATEMI GHOMI, S. M. T. Hybrid flowshop scheduling with machine and resource-dependent processing times. *Applied Mathematical Modelling*, v. 35, n. 3, p.1107-1123, 2011.

BOTTA-GENOULAZ, V. Hybrid flow shop scheduling with precedence constraints and time lags to minimize maximum lateness. *International Journal of Production Economics*, v. 64, n. 1-3, p. 101-111, 2000.

CHEN, Y. -Y. et al. A hybrid approach based on the variable neighborhood search and particle swarm optimization for parallel machine scheduling problems – a case study for solar cell industry. *International Journal of Production Economics*, v. 141, p. 66-78, 2013.

CONWAY, R. W.; MAXWELL, W. L.; MILLER, L. W. *Theory of scheduling*. Massachussets: Addison-Wesley, 1967.

DAS, S. R.; GUPTA, J. N. D.; KHUMAWALA, B. M. A saving index heuristic algorithm for flowshop scheduling with sequence dependent set-up times. *Journal of Operational Research Society*, v. 46, p. 1365-1373, 1995.

FATTAHI, P. et al. A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations. *Applied Mathematical Modelling*, in press, 2013.

GÓMEZ-GASQUET, P.; ANDRÉS, C.; LARIO, F.-C. An agent-based genetic algorithm for hybrid flowshops with sequence dependent setup times to minimize makespan. *Expert Systems with Applications*, v. 39, n. 9, p. 8095-8107, 2012.

GUPTA, J. N. D.; TUNC, E. A. Scheduling a two-stage hybrid flowshop with separable setup and removal times. *European Journal of Operational Research*, v. 77, p. 415-428, 1994.

HEKMATFAR, H.; FATEMI GHOMI, S. M. T.; KARIMI, B. Two stage reentrant hybrid flow shop with setup times and the criterion of minimizing makespan. *Applied Soft Computing*, v. 11, n. 8, p.4530-4539, 2011.

HUANG, W.; LI, S. A two-stage hybrid flowshop with uniform machines and setup times. *Mathematical and Computer Modeling*, v. 27, n. 2, p. 27-45, 1998.

JUNG, C. F. *Metodologia para pesquisa & desenvolvimento*: aplicada a novas tecnologias, produtos e processos. Rio de Janeiro: Axcel Books, 2004.

KIS, T.; PESCH, E. A review of exact solution methods for the non-preemptive multiprocessor flowshop problem. *European Journal of Operational Research*, v. 164, p. 592-608, 2005.

LI, S. A hybrid two-stage flowshop with part family, batch production, and major and minor set-ups. *European Journal of Operational Research*, v. 102, p. 142-156, 1997.

LINN, R.; ZHANG, W. Hybrid flow shop scheduling: a survey. *Computers & Industrial Engineering*, v. 37, n. 1-2, p. 57-61, 1999.

LOGENDRAN, R.; CARSON, S.; HANSON, E. Group scheduling in flexible flow shops. *International Journal of Production Economics*, v. 96, p. 143-155, 2005.

LOW, C. Simulated anneling heuristic for flow shop scheduling problems with unrelated parallel machines. *Computers & Operations Research*, v. 32, p. 2013-2025, 2005.

MARTINS, R. A. Abordagens quantitativa e qualitativa. In: MIGUEL, P. A. C. (Org.). *Metodologia de pesquisa em Engenharia de Produção e Gestão de Operações*. Rio de Janeiro: Elsevier, 2010. p. 45-61, cap. 3.

NAKANO, D. Métodos de pesquisa adotados na Engenharia de Produção e Gestão de Operações. In: MIGUEL, P. A. C. (Org.). *Metodologia de pesquisa em Engenharia de Produção e Gestão de Operações*. Rio de Janeiro: Elsevier, 2010. p. 63-72, cap. 4.

PINEDO, M. *Scheduling*: theory, algorithms and systems. New Jersey: Prentice-Hall, 2010.

QUADT, D.; KUHN, H. A taxonomy of flexible flow line scheduling procedures. *European Journal of Operational Research*, v. 178, p. 686-698, 2007.

RAJENDRAN, C.; ZIEGLER, H. A heuristic for scheduling to minimize the sum of weighted flowtime of jobs in a flowshop with sequence-dependent setup times of jobs. *Computers & Industrial Engineering*, v. 33, n. 1-2, p. 281-284, 1997.

RIBAS, I.; LEISTEN, R.; FRAMIÑAN, J. M. Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers & Operations Research*, v. 37, p. 1439-1454, 2010.

RÍOS-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.

RÍOS-MERCADO, R. Z.; BARD, J. F. An enhanced TSP-based heuristic for makespan minimization in a flow shop with setup times. *Journal of Heuristics*, v. 5, p. 53-70, 1999.

RUIZ, R.; VÁZQUEZ-RODRÍGUEZ, J. A. The hybrid flow shop scheduling problem. *European Journal of Operational Research*, v. 205, p. 1-18, 2010.

SIMONS JR., J. Heuristics in flow shop scheduling with sequence dependent setup times. *Omega – The International Journal of Management Science*, v. 20, n. 2, p. 215-225, 1992.

VAIRAKTARAKIS, G. Flexible hybrid flowshops. In: LEUNG, J.Y-T. *Handbook of Scheduling*: algorithms, models, and performance analysis. San Diego: Chapman & Hall/CRC Press, 2004, p. 5.1-5.33.

VIGNIER, A.; BILLAUT, J. C.; PROUST, C. Les problèmes d'ordonnancement de type flow-shop hybride: état de l'art. *RAIRO – Recherche Opérationnelle*, v. 33, n. 2, p. 117-183, 1999.

WANG, H. Flexible flow shop scheduling: optimum, heuristic and artificial intelligence solutions. *Expert Systems*, v. 22, n. 2, p. 78-85, 2005.

WENG, M. X.; LU, J.; REN, H. Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective. *International Journal of Production Economics*, v. 70, n. 3, p. 215-226, 2001.

WENG, W.; WEI, X.; FUJIMURA, S. Dynamic routings strategies for JIT production in hybrid flow shops. *Computers & Operations Research*, v. 39, p. 3316-3324, 2012.

Recebido em 25 jul. 2013 / aprovado em 4 out. 2013

Para referenciar este texto
FUCHIGAMI, H. Y.; MOCCELLIN, J. V. Estudo da influência da programação do primeiro estágio em *flow shop* híbridos com tempos de *setup* explícitos. *Exacta – EP*, São Paulo, v. 11, n. 2, p. 149-160, 2013.