# UM MODELO DE PROGRAMAÇÃO INTEIRA MISTA PARA A PROGRAMAÇÃO DA PRODUÇÃO EM *FLOWSHOP* HÍBRIDO COM *BUFFERS* LIMITADOS

**Pedro Luis Miranda Lugo**
Universidade Federal de São Carlos – Departamento de Engenharia de Produção
pmiranda@ufscar.br

**Rodolfo Florence Teixeira Jr**
Universidade Federal de São Carlos – *Campus* Sorocaba – Departamento de Engenharia de Produção
rodolfo.florence@ufscar.br

**RESUMO**

Esta pesquisa aborda o problema de programação da produção em sistemas *Flowshop* híbrido. Algumas restrições comumente encontradas em sistemas de produção reais, como máquinas paralelas não relacionadas, *buffers* limitados, tempos de preparação dependentes da sequência (antecipatórios e não antecipatórios), elegibilidade de máquinas, tempos de transporte e tempos de liberação das máquinas, são consideradas. O critério de otimização é o *makespan*, cuja minimização está diretamente relacionada com a utilização eficiente dos recursos de produção. Um modelo de programação inteira mista é proposto e resolvido através do *solver* comercial CPLEX. Os resultados da avaliação computacional indicam que o modelo é viável somente para resolver instâncias de até nove tarefas e cinco estações.

**PALAVARAS-CHAVE: *Flowshop* híbrido, programação da produção, *buffers* limitados.**

## 1. Introdução

Em um *Flowshop* um conjunto de tarefas deve ser processado através de múltiplas estações na mesma ordem, desde a primeira até a última estação, e cada estação tem unicamente uma máquina. Porém, como destacado por Ribas, Leisten e Framiñan (2010), em algumas indústrias a necessidade de incrementar ou equilibrar a capacidade das estações tem levado à duplicação de máquinas. Em outras, a crescente demanda de produtos customizados tem desencadeado a necessidade de comprar máquinas adicionais para algumas estações com o objetivo de dedicá-las à produção deste produtos. Em qualquer caso, a aquisição das novas máquinas não implica a substituição ou eliminação do equipamento existente, o que leva a coexistência de várias máquinas, possivelmente diferentes, em várias estações do processo de produção.

Do ponto de vista teórico, esta nova configuração de produção, conhecida como *Flowshop* Híbrido (HFS), pode ser considerada como a combinação de dois problemas particulares de programação da produção: o problema de máquinas paralelas e o problema do *Flowshop*. No problema de máquinas paralelas a decisão-chave é a alocação de tarefas a máquinas, enquanto no *Flowshop* a decisão-chave é o sequenciamento das tarefas através do chão de fábrica. Portanto, no HFS as decisões-chave são designar e programar as tarefas às máquinas em cada estação, isto é, determinar a ordem na qual as tarefas serão processadas nas diferentes máquinas de cada estação, com o objetivo de minimizar um ou vários critérios determinados.

O estudo deste problema é de grande relevância prática, dado que este tipo de configuração produtiva pode ser encontrado em diversos setores de manufatura, como na indústria petroquímica (Deal, Yang, & Hallquist, 1994), indústria automotiva (Agnetis et al., 1997), indústria eletrônica (Liu & Chang, 2000), indústria têxtil (Grabowski & Pempera, 2000),

manufatura de etiquetas autoadesivas (Lin & Liao, 2003), indústria de revestimentos cerâmicos (Ruiz, Şerifoğlu, & Urlings, 2008), entre outras.

## 2. Descrição do problema

Formalmente, um conjunto $N$ de tarefas, $N = \{1,...,n\}$, deve ser processado em um conjunto $M$ de estações, $M = \{1,...,m\}$. Em cada estação $i$, $i \in M$, há um conjunto $M_i = \{1,...,m_i\}$ de processadores paralelos não relacionados. Os processadores podem ser máquinas ou *buffers*, sendo os últimos considerados máquinas especiais com tempos de processamento e preparação iguais a zero. $E_{ij}$ denota o conjunto de processadores capazes de processar a tarefa $j$, $j \in N$, na estação $i$. O parâmetro $rm_{il}$ indica o tempo de liberação do processador $l$, $l \in M_i$, na estação $i$. Cada tarefa $j$ deve ser processada por exatamente um dos processadores paralelos em cada estação $i$. O tempo de processamento da tarefa $j$ no processador $l$ da estação $i$ é denotado como $p_{ilj}$. O parâmetro $S_{iljk}$ denota o tempo de preparação entre as tarefas $j$ e $k$, $k \in N$, no processador $l$ da estação $i$. Adicionalmente, o parâmetro binário $A_{iljk}$ indica se a preparação entre as tarefas $j$ e $k$ no processador $l$ da estação $i$ é antecipatória ou não antecipatória. Operações de transporte entre estações devem ser feitas antes de iniciar o processamento de qualquer tarefa, assim $t_{iq}$ indica o tempo de transporte entre as estações $i$ e $q, q \in M$. O critério de otimização é o $C_{\max}$, tal que $C_{\max} = \max_{j \in N} \{C_j\}$ e $C_j$ indica o tempo de finalização da tarefa $j$ na última estação.

## 3. Modelo Matemático

Nesta seção apresenta-se um modelo de programação inteira mista, baseado no modelo de Ruiz et al. (2008), para o HFS abordado nesta pesquisa. O modelo é testado em um *benchmark* de 1.728 instâncias e os resultados são utilizados para determinar a viabilidade e desempenho do modelo proposto como ferramenta de solução.

**Índices e Conjuntos:**

$j, k, h$ = tarefas;

$i, q$ = estações;

$l$ = processadores;

$N$ = conjunto de tarefas;

$M$ = conjunto de estações;

$M_i$ = conjunto de processadores paralelos na estação *i*;

$E_{ij}$ = conjunto de processadores elegíveis para a tarefa *j* na estação *i*;

$G_{il}$ = conjunto de tarefas que podem ser processadas no processador $l$ da estação *i*:

**Parâmetros:**

$r_{il}$ = data de liberação do processador $l$ da estação *i*;

$p_{ilj}$ = tempo de processamento da tarefa $j$ no processador *l* da estação *i*;

$S_{iljk}$ = tempo de setup entre as tarefas $j$ e $k$ no processador *l* da estação *i*;

$t_{iq}$ = tempo de transporte entre as estações *i* e *q*;

$A_{iljk} = \begin{cases} 1, \text{ se o setup entre as tarefas } j \text{ e } k \text{ no processador } l \text{ da estação } i \text{ é anticipatório} \\ 0, \text{ caso contrário} \end{cases}$

$FS(LS) =$ Primeira (última) estação do sistema.

**Variáveis de Decisão:**

$X_{iljk} = \begin{cases} 1, \text{ se a tarefa } j \text{ precede a tarefa } k \text{ no processador } l \text{ da estação } i \\ 0, \text{ caso contrário} \end{cases}$

$C_{ij} =$ tempo de finalização da tarefa $j$ na estação $i$;

$d_{ij} =$ tempo de saída da tarefa $j$ da estação $i$;

$C_{\max} =$ *Makespan.*

$$\min C_{\max} \tag{3.1}$$

$$\sum_{\substack{j\in\{N,0\}\\ j\neq k}} \sum_{l\in E_{ij}\cap E_{ik}} X_{iljk} = 1, \qquad k\in N, i\in M \tag{3.2}$$

$$\sum_{\substack{j\in N\\ j\neq k}} \sum_{l\in E_{ij}\cap E_{ik}} X_{ilkj} \leq 1, \qquad k\in N, i\in M \tag{3.3}$$

$$\sum_{\substack{h\in\{G_{il},0\}\\ h\neq j, h\neq k}} X_{ilhj} \geq X_{iljk}, \qquad j,k\in N,\ j\neq k, i\in M, l\in E_{ij}\cap E_{ik} \tag{3.4}$$

$$\sum_{l\in E_{ij}\cap E_{ik}} (X_{iljk} + X_{ilkj}) \leq 1, \qquad j\in N, k = j+1,...,n, i\in M \tag{3.5}$$

$$\sum_{k\in G_{il}} X_{il0k} \leq 1, \qquad i\in M, l\in M_i \tag{3.6}$$

$$C_{i0} = 0, \qquad i\in M \tag{3.7}$$

$$C_{ik} + B(1-X_{iljk}) \geq rm_{il} + (1-A_{iljk})S_{iljk} + p_{ilk}, \quad k\in N, i\in M, l\in E_{ik}, \ j\in\{G_{il},0\},\ j\neq k \tag{3.8}$$

$$C_{ik} + B(1-X_{iljk}) \geq d_{ij} + S_{iljk} + p_{ilk}, \quad k\in N, i\in M, l\in E_{ik}, \ j\in\{G_{il},0\},\ j\neq k \tag{3.9}$$

$$C_{ik} + B(1-X_{iljk}) \geq C_{i-1,k} + t_{i-1,i} + (1-A_{iljk})S_{iljk} + p_{ilk}, \quad k\in N, i\in\{M\setminus FS\}, \ l\in E_{ik},\ j\in\{G_{il},0\},\ j\neq k \tag{3.10}$$

$$d_{i-1,k} = C_{ik} - \sum_{\substack{j\in\{N,0\}\\ j\neq k}} \sum_{l\in E_{ij}\cap E_{ik}} X_{iljk}\cdot p_{ilk} - \sum_{\substack{j\in\{N,0\}\\ j\neq k}} \sum_{l\in E_{ij}\cap E_{ik}} X_{iljk}(1-A_{iljk})S_{iljk} - t_{i-1,i}, \quad k\in N, i\in\{M\setminus FS\} \tag{3.11}$$

$$d_{ik} \geq C_{ik}, \qquad k\in N, i\in\{M\setminus LS\} \tag{3.12}$$

$$d_{ik} = C_{ik}, \qquad k\in N, i = LS \tag{3.13}$$

$$C_{\max} \geq C_{ik}, \qquad k\in N, i = LS \tag{3.14}$$

$$X_{iljk}\in\{0,1\}, \qquad j\in\{N,0\}, k\in N, j\neq k, i\in M, l\in E_{ij}\cap E_{ik} \tag{3.15}$$

$$C_{ij}, d_{ij} \geq 0, \qquad j\in N, i\in M \tag{3.16}$$

A função objetivo (3.1) visa à minimização do *makespan*. As restrições (3.2) asseguram que cada tarefa $k$ deve ser precedida por uma e somente uma tarefa $j$ em apenas um processador $l$ em cada estação $i$. As restrições (3.3) indicam que cada tarefa $k$ deve ter, no máximo, um sucessor $j$. O conjunto de restrições (3.4) garante que se a tarefa $j$ é predecessora de alguma tarefa $k$ em algum processador $l$ da estação $i$, então a tarefa $j$ deve ter um predecessor $h$ no mesmo processador $l$. O restrições (3.5) evitam a ocorrência de precedência cruzada. As restrições (3.6) indicam que a tarefa fictícia zero pode ser predecessora de, no máximo, uma

tarefa $k$ em cada processador $l$ em cada estação $i$. O conjunto de restrições (3.7) garante que a tarefa fictícia zero seja finalizada no tempo zero em cada estação $i \in M$. As restrições (3.8) indicam que se a tarefa $k$ é designada ao processador $l$ da estação $i$, então seu processamento não pode começar até que o processador $l$ seja liberado pelo programação de produção anterior. O conjunto de restrições (3.9) indica que o processamento da tarefa $k$ no processador $l$ da estação $i$ não pode começar até que a tarefa predecessora $j$ tenha saído da estação $i$ e a correspondente preparação do processador $l$ tenha sido feita. As restrições (3.10) garantem que cada tarefa $k$ seja processada subsequentemente em cada estação $i$. As restrições (3.11) calculam o tempo de saída da tarefa $k$ da estação $i$. As restrições (3.12) indicam que o tempo de saída da tarefa $k$ da estação $i$ não pode ser menor que seu tempo de finalização nesta estação. O conjunto de restrições (3.13) garante que a tarefa $k$ sai do sistema assim que finalizar seu processamento na última estação. As restrições (3.14) define o *makespan* e, finalmente, as restrições (3.15) e (3.16) definem o domínio das variáveis de decisão.

## 4. Avaliação Computacional

O modelo de programação inteira mista da seção 3 foi implementado no *GAMS* e um conjunto de 1.728 instâncias foi resolvido utilizando o *CPLEX 12* em um *notebook* com processador Intel Core i5 2.67 GHz com 4 Gigabytes de memória RAM. Devido à quantidade de instâncias, um tempo limite de execução de 3.600 segundo foi imposto para cada uma delas.

Na Tabela 1 são apresentados os resultados consolidados do modelo. Cada célula representa a média para cada tamanho de instância. Assim, são apresentadas a percentagem de instâncias cuja solução ótima foi encontrada (%Ótimo); a percentagem de instâncias para as quais uma solução inteira factível foi encontrada (%IntSol) e a percentagem de instâncias para as quais nenhuma solução inteira factível foi encontrada (%NãoSol).

| | | *m* | | |
|---|---|---|---|---|
| *n* | | 3 | 5 | 7 |
| 5 | %Ótimo | 100,00 | 100,00 | 100,00 |
| | %IntSol | 0,00 | 0,00 | 0,00 |
| | %NãoSol | 0,00 | 0,00 | 0,00 |
| 7 | %Ótimo | 100,00 | 97,92 | 86,81 |
| | %IntSol | 0,00 | 2,08 | 9,72 |
| | %NãoSol | 0,00 | 0,00 | 3,47 |
| 9 | %Ótimo | 94,44 | 52,08 | 22,92 |
| | %IntSol | 5,56 | 46,53 | 70,83 |
| | %NãoSol | 0,00 | 1,39 | 6,25 |
| 11 | %Ótimo | 40,97 | 13,19 | 1,39 |
| | %IntSol | 59,03 | 81,94 | 79,17 |
| | %NãoSol | 0,00 | 4,86 | 19,44 |

**Tabela 1: Resultados consolidados do modelo de programação inteira mista**

O *gap* médio observado é relativamente alto, entre 22,25% e 56,67%, e tende a ser maior com o aumento do número de tarefas e estações. Este comportamento era esperado porque, como mostrado pela Tabela 1, quanto maior número de tarefas e estações, maior a dificuldade teve o *CPLEX* para resolver as instâncias.

A Figura 1 ilustra o comportamento do tempo médio utilizado pelo *solver* em função do número de tarefas e estações. Note que, para instâncias além de nove tarefas e cinco estações, atingir uma solução ótima é inviável do ponto de vista prático, dado a quantidade de tempo requerida.

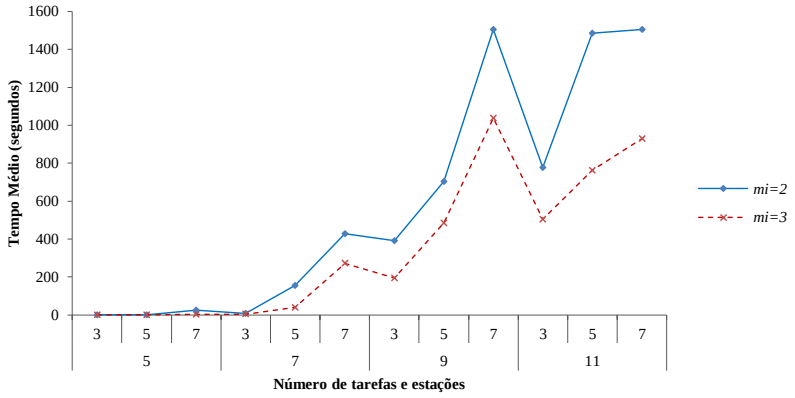


**Figura 1: Tempo Médio utilizado pelo solver para encontrar uma solução ótima**

## 5.Conclusões e perspectivas futuras

Nesta pesquisa um modelo de programação inteira mista foi apresentado para resolver o problema de programação da produção em HFS com máquinas paralelas não relacionadas, tempos de preparação dependentes da sequência (antecipatórios e não antecipatórios), tempos de transporte entre estações, *buffers* limitados e elegibilidade de máquinas, visando minimizar o *makespan*. O modelo foi testado em um conjunto de 1.728 instâncias com tempo limite de 3.600 segundos e os resultados encontrados indicam que o modelo é computacionalmente viável para instância de até nove tarefas e cinco estações. Em termos gerais, o CPLEX encontrou ao menos uma solução ótima para 1.166 instâncias. Para 511 instâncias, encontrou ao menos uma solução inteira factível e, finalmente, para as 51 instâncias restantes nenhuma solução inteira factível foi encontrada depois de 3.600 segundos.

Como pesquisa futura, pretende-se implementar e utilizar métodos meta-heurísticos para resolver o problema estudado. Estes métodos precisam maior tempo computacional que outros métodos mais simples, como as heurísticas, mas em contrapartida espera-se encontrar soluções de melhor qualidade para instâncias de qualquer tamanho.

## Referências

Agnetis, A., Pacifici, A., Rossi, F., Lucertini, M., Nicoletti, S., Nicolò, F., … Pesaro, E. (1997). Scheduling of flexible flow lines in an automobile assembly plant. *European Journal of Operational Research*, *97*(2), 348–362. doi:10.1016/S0377-2217(96)00203-2

Deal, D. E., Yang, T., & Hallquist, S. (1994). Job scheduling in petrochemical production: Two-stage processing with finite intermediate storage. *Computers and Chemical Engineering*, *18*(4), 333–344.

Grabowski, J., & Pempera, J. (2000). Sequencing of jobs in some production system. *European Journal of Operational Research*, *125*(3), 535–550. doi:10.1016/S0377-2217(99)00224-6

Lin, H.-T., & Liao, C.-J. (2003). A case study in a two-stage hybrid flow shop with setup time and dedicated machines. *International Journal of Production Economics*, *86*(2), 133–143. doi:10.1016/S0925-5273(03)00011-2

Liu, C.-Y., & Chang, S.-C. (2000). Scheduling flexible flow shops with sequence-dependent setup effects. *IEEE Transactions on Robotics and Automation*, *16*(4), 408–419. doi:10.1109/70.864235

Ribas, I., Leisten, R., & Framiñan, J. M. (2010). Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers & Operations Research*, *37*(8), 1439–1454. doi:10.1016/j.cor.2009.11.001

Ruiz, R., Şerifoğlu, F., & Urlings, T. (2008). Modeling realistic hybrid flexible flowshop scheduling problems. *Computers & Operations Research*, *35*(4), 1151–1175. doi:10.1016/j.cor.2006.07.014