Estudo da influência da relação entre os tempos de processamento e de *setup* em sistemas *flow shop* híbridos

Hélio Yochihiro Fuchigami (USP-EESC) heliofuchigami@yahoo.com.br
João Vitor Moccellin (USP-EESC) jvmoccel@sc.usp.br

Resumo
*O foco deste trabalho é o estudo da influência da relação entre as ordens de grandeza dos tempos de processamento das tarefas e os tempos de preparação das máquinas (*setup*) em ambientes* flow shop *com máquinas múltiplas. No problema considerado, os tempos de preparação são assimétricos e dependentes da seqüência de execução das tarefas. A função-objetivo é a minimização da duração total da programação (*makespan*). Os resultados dos métodos heurísticos construtivos propostos são comparados com seis relações de tempos de processamento e de* setup*.*
Palavras-chave: programação da produção, flow shop *híbrido,* setup *dependente, métodos heurísticos.*

1 Introdução

O objetivo básico deste trabalho é avaliar a influência da relação entre as ordens de grandeza dos tempos de processamento das tarefas e os tempos de preparação das máquinas (*setup*) na programação da produção em ambientes *flow shop* híbridos, em que os tempos de *setup* são assimétricos e dependentes da seqüência de execução das tarefas. O critério de desempenho é a minimização do *makespan* (duração total da programação).

Muitas pesquisas em programação da produção desconsideram os tempos de preparação das máquinas ou então os incluem nos tempos de processamento das tarefas. Isto simplifica a análise das aplicações, porém afeta a qualidade da solução quando tais tempos têm uma variabilidade relevante em função da ordenação das tarefas nas máquinas, como é o caso de indústrias químicas, gráficas, têxteis e de papel, caracterizadas por sistemas com amplo *mix* de produtos.

2 Programação de operações em sistemas *flow shop* híbridos com tempos de preparação dependentes da seqüência

Huang & Li (1998) apresentaram um ambiente *flow shop* híbrido com dois estágios e tempos de *setup* dependentes da seqüência de famílias de produtos, baseado em um problema real. O primeiro estágio continha somente uma máquina e o segundo, várias máquinas uniformes. O objetivo foi a minimização do *makespan*. Para a solução do problema, foram construídas duas heurísticas e definidas oito regras de seqüenciamento. Também foi apresentado um modelo matemático de programação linear considerando o *trade-off* entre o custo de máquinas rápidas e o *makespan*.

Um estudo de caso para o problema *flow shop* híbrido com dois estágios e tempos de *setup* dependentes da seqüência somente no primeiro estágio, grupos de máquinas idênticas no segundo estágio e dois prazos de entrega (*due dates*) foi realizado por Lin & Liao (2003). O objetivo foi minimizar o atraso máximo ponderado. O resultado da heurística proposta foi comparado com a solução ótima de problemas de pequeno porte (7 a 10 tarefas) e com a solução do método de programação utilizado na fábrica. A heurística ordena as tarefas no

primeiro estágio minimizando o tempo total de *setup* e emprega a regra *First In First Out* (FIFO) no segundo estágio.

Fuchigami (2005) estudou o problema de programação em *flow shop* híbrido com tempos de *setup* assimétricos e dependentes da seqüência de processamento das tarefas. Foram propostos e avaliados quatro métodos heurísticos construtivos.

Esse é o ambiente de produção tratado neste trabalho. Como pode ser observado pelo exame da literatura, existem poucos trabalhos reportados até o momento.

3 Métodos heurísticos construtivos propostos

O ambiente de produção *flow shop* híbrido é ilustrado na figura 1.

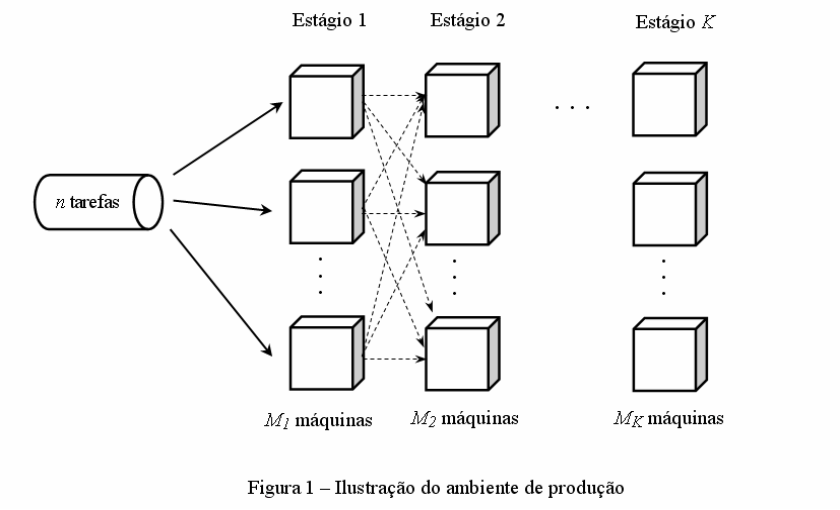

Figura 1 – Ilustração do ambiente de produção

O problema consiste em programar um conjunto de n tarefas, definido como $J = \{J_1, ..., J_n\}$, onde cada tarefa possui necessariamente uma única operação em cada estágio de produção. As operações das tarefas devem ser realizadas seqüencialmente e passam por todos os estágios. O objetivo do problema é minimizar o *makespan* como medida de desempenho.

Para obter a solução desse problema de programação da produção, foram desenvolvidos e avaliados quatro métodos heurísticos construtivos com base em algoritmos reportados na literatura para solução de problemas *flow shop* permutacional (SIMONS JR., 1992) e máquinas paralelas (WENG, LU & REN, 2001) no ambiente cujo tempo de *setup* é dependente da seqüência de execução das tarefas.

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. Os estágios seguintes consistem de $K - 1$ 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.

Os algoritmos dos quatro métodos de solução são apresentados a seguir.

Procedimento 1 – estabelecimento de ordenação inicial

O Procedimento 1 define uma ordenação inicial de tarefas e utiliza duas regras de prioridade: a *Longest Processing Time* (LPT) e um método baseado no algoritmo TOTAL de Simons Jr. (1992). Desta forma, as regras de prioridade deste procedimento são denominadas LPT e TOTAL, respectivamente.

A regra LPT considera a soma dos tempos de processamento de cada tarefa em todos os estágios e faz o seqüenciamento das tarefas pela ordem não-crescente destas somas.

A regra de prioridade TOTAL baseia-se na heurística de Simons Jr. (1992) para programação de *flow shop* permutacional com tempos de *setup* dependentes da seqüência.

O cálculo das datas de término das tarefas J_i em um determinado estágio k é feito utilizando a seguinte expressão:

$$C_{E_{ik}} = \min_{m=1}^{M_k} \left\{ \max \left\{ C_{u_m k} + s_{u_m ik} ; C_{i,k-1} \right\} + p_{ik} \right\} \qquad (3.1)$$

com $i = 1, ..., n$ e $k = 1, ..., K$.

$C_{E_{ik}}$ representa a menor data de término da tarefa J_i no estágio k, M_k é o número de máquinas no estágio k, u_m é o índice da última tarefa alocada na máquina m (J_{u_m}), $C_{u_m k}$ denota a data de término da tarefa J_{u_m} no estágio k, $s_{u_m ik}$ é o tempo de *setup* da tarefa J_i no estágio k após o processamento de J_{u_m}, $C_{i,k-1}$ denota a data de término da tarefa J_i no estágio k-1 e p_{ik} é o tempo de processamento da tarefa J_i no estágio k.

ALGORITMOS DO PROCEDIMENTO 1 – REGRAS LPT E TOTAL

Seja J um conjunto de n tarefas a serem programadas, J' o conjunto das tarefas ainda não programadas, $J_{[j]}$ a tarefa que ocupa a j-ésima posição de J', M o número total de máquinas considerando todos os estágios de produção ($M = \sum_{k=1}^{K} M_k$, onde M_k é o número de máquinas no estágio k) e σ_m o subconjunto de tarefas alocadas à máquina m. Seja r_i a data de liberação da tarefa J_i e SRD (*Shortest Release Date*) a ordenação das tarefas segundo os valores não-decrescentes de r_i.

PASSO 1 (INICIALIZAÇÃO)

$\sigma_m = \varnothing$, para $m = 1, 2, ..., M$

$J' \leftarrow J$

Ordene as tarefas do conjunto J' de acordo com a regra de prioridade (LPT ou TOTAL). Vá para o PASSO 3.

PASSO 2

$J' \leftarrow J$

Ordene as tarefas do conjunto J' de acordo com a ordenação SRD, considerando as datas de término das tarefas no estágio anterior como as datas de liberação do estágio atual.

PASSO 3

A primeira tarefa de *J'* será alocada à máquina *x*, cuja programação possuirá a data mais cedo de término.

PASSO 4 (ATUALIZAÇÃO DOS CONJUNTOS E TESTE DE PARADA)

$\sigma_x \leftarrow \sigma_x \cup \{ J'_{[1]} \}$

$J' \leftarrow J' - \{ J'_{[1]} \}$

Se $J' \neq \varnothing$, vá para o PASSO 3.
Caso contrário, programação do estágio concluída.
Se não for o último estágio, vá para o PASSO 2.
Caso contrário PARE, programação concluída. Saída: σ_m para $m = 1, 2... M$.

Procedimento 2 – sem estabelecimento de ordenação inicial

O Procedimento 2 não utiliza ordenação inicial para as tarefas e uma das regras de alocação emprega uma adaptação do melhor entre sete algoritmos para máquinas paralelas com *setup* dependente apresentados por Weng, Lu & Ren (2001). A outra regra de alocação deste procedimento também utiliza a adaptação do algoritmo citado, mas quando todas as tarefas estiverem liberadas respeita a ordenação pela regra LPST (*Longest Processing-Setup Time*), envolvendo a soma do tempo de processamento com o tempo de *setup*. As regras de alocação deste procedimento são designadas SCT (*Shortest Completion Time*) e SCT/LPST, respectivamente.

Como não há estabelecimento de uma ordenação inicial de tarefas, a regra SCT estabelece que a alocação é feita por meio de uma expressão que analisa todas as possibilidades de combinação tarefa-máquina e escolhe o par com a data de término mais cedo. Em cada estágio *k*, a seguinte expressão é calculada:

$$C_{E_{ik}} = \min\left\{ C_{u_m k} + s_{u_m ik} + p_{ik} \mid m \in \mu_k \right\} \qquad (3.2)$$

com $i = 1,..., n$ e $k = 1, ..., K$, onde μ_k é o conjunto dos índices das máquinas do estágio *k* e as variáveis são as mesmas da expressão 3.1.

A regra de alocação SCT/LPST segue o mesmo procedimento da regra SCT, porém quando todas as tarefas estiverem liberadas, respeita a ordenação pela regra LPST. Considerando a máquina de menor carga, a tarefa alocada é aquela com o maior valor da soma dos tempos de *setup* e de processamento $(s_{u_m ik} + p_{ik})$.

ALGORITMO DO PROCEDIMENTO 2 – REGRA SCT

PASSO 1 (INICIALIZAÇÃO)

$\sigma_m = \varnothing$, para $m = 1, 2, ..., M$

PASSO 2

$J' \leftarrow J$

PASSO 3

A partir da primeira data com tarefas liberadas, calcule a data de término de cada uma das tarefas de *J'* já liberadas, em cada uma das máquinas do estágio, ou seja, analise

todas as possibilidades de associação tarefa-máquina, e escolha o par tarefa J_i e máquina x com a data de término mais cedo.

PASSO 4 (ATUALIZAÇÃO DOS CONJUNTOS E TESTE DE PARADA)

$\sigma_x \leftarrow \sigma_x \cup \{J_i\}$
$J' \leftarrow J' - \{J_i\}$
Se $J' \neq \varnothing$, vá para o PASSO 3.
Caso contrário, programação do estágio concluída.
Se não for o último estágio, vá para o PASSO 2.
Caso contrário PARE, programação concluída. Saída: σ_m para $m = 1, 2... M$.

ALGORITMO DO PROCEDIMENTO 2 – REGRA SCT/LPST

PASSO 1 (INICIALIZAÇÃO)

$\sigma_m = \varnothing$, para $m = 1, 2, ..., M$

PASSO 2

$J' \leftarrow J$

PASSO 3

Se todas as tarefas estiverem liberadas, vá para o PASSO 4.

Caso contrário, a partir da primeira data com tarefas liberadas, calcule a data de término de cada uma das tarefas de J' já liberadas, em cada uma das máquinas do estágio, ou seja, analise todas as possibilidades de associação tarefa-máquina, e escolha o par tarefa J_i e máquina x com a data de término mais cedo. Vá para o PASSO 5.

PASSO 4

Na máquina de menor carga, associe a tarefa J_i com o maior valor de $(s_{u_m ik} + p_{ik})$, ou seja, observe a regra LPST.

PASSO 5 (ATUALIZAÇÃO DOS CONJUNTOS E TESTE DE PARADA)

$\sigma_x \leftarrow \sigma_x \cup \{J_i\}$
$J' \leftarrow J' - \{J_i\}$
Se $J' \neq \varnothing$, vá para o PASSO 3.
Caso contrário, programação do estágio concluída.
Se não é o último estágio, vá para o PASSO 2.
Caso, contrário PARE, programação concluída. Saída: σ_m para $m = 1, 2... M$.

Observação: nos algoritmos dos dois procedimentos, considera-se que uma máquina qualquer, ao receber a primeira tarefa, já esteja preparada para sua execução.

4 Experimentação computacional

4.1 Delineamento do experimento

O foco da experimentação computacional foi a análise da influência da relação entre as ordens de grandeza dos tempos de processamento e de *setup* das tarefas, denotada por $\boldsymbol{O(p_i)/O(s_{ij})}$, no desempenho dos métodos desenvolvidos. Por este motivo, foram definidas seis relações, como consta na Tabela 1.

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

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

A geração dos tempos de processamento e de *setup* dentro de cada intervalo foi feita de forma aleatória com distribuição uniforme. Os limites dos intervalos das relações foram definidos e padronizados com base em trabalhos reportados na literatura (SIMONS JR., 1992; RAJENDRAN & ZIEGLER, 1997; RÍOS-MERCADO & BARD, 1998 e 1999; DAS, GUPTA & KHUMAWALA, 1995; WENG, LU & REN, 2001).

Na experimentação computacional foram testados 14.400 problemas, divididos em 144 classes definidas pelo número de tarefas $n \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120\}$, número de estágios de produção $K \in \{4, 7\}$ e pelas seis relações entre as ordens de grandeza dos tempos de processamento e de *setup* ($O(p_i)/O(s_{ij})$). Para cada classe, foram gerados aleatoriamente 100 problemas, com o objetivo de reduzir o erro amostral.

Em cada estágio, o número de máquinas paralelas idênticas varia de 2 a 5, 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 classes.

Para simplificação da notação, o Procedimento 1 com a regra de prioridade LPT foi denotado como "11" e com a regra de prioridade TOTAL como "12". O Procedimento 2 com a regra de alocação SCT foi denominado "21" e com a regra SCT/LPST foi indicado como "22".

4.2 Análise dos resultados

Tendo em vista o objetivo básico deste trabalho, os resultados obtidos na experimentação computacional foram analisados por meio da porcentagem de sucesso 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.

Como não foram encontrados na literatura métodos heurísticos construtivos para programação do ambiente tratado neste trabalho, os algoritmos desenvolvidos foram comparados entre si.

Na totalidade dos problemas resolvidos, o método 21 apresentou a melhor solução 7.130 vezes num total de 14.400 problemas, correspondendo a 49,5% de sucesso. Em segundo lugar, o método 12 obteve a melhor solução em 4.531 problemas, equivalente a 31,5% de sucesso. Em seguida, o método 11 forneceu a melhor solução 2581 vezes, ou seja, atingiu 17,9% de sucesso. E, por último, 232 problemas tiveram a melhor solução pelo método 22, com 1,6% de sucesso.

A Tabela 2 apresenta uma comparação geral do desempenho dos métodos em todas as relações $O(p_i)/O(s_{ij})$ e para 4 e 7 estágios com base na porcentagem de sucesso. Essa tabela

recomenda um determinado método em função do número de tarefas, número de estágios e relação $O(p_i)/O(s_{ij})$.

	K = 4						K = 7					
n	RI	RII	RIII	RIV	RV	RVI	RI	RII	RIII	RIV	RV	RVI
10	21	21	21	21	21	11	21	11	21	21	21	21
20	21	21	21	21	21	21	21	21	21	21	21	21
30	21	21	21	21	21	21	21	21	21	21	21	21
40	21	21	21	21	21	21	21	21	21	21	21	21
50	12 21	21	21	21	12	21	21	21	21	21	12	21
60	21	21	21	12	12	21	21	21	21	12	12	21
70	12	21	21	12	12	21	12	21	21	12	12	21
80	12	21	21	12	12	21	12	21	21	12	12	21
90	12	21	21	12	12	21	12	21	21	12	12	21
100	12	21	21	12	12	21	12	21	21	12	12	21
110	12	21	21	12	12	21	12	12	21	12	12	21
120	12	21	21	12	12	21	12	21	21	12	12	21

Tabela 2 – Resumo do desempenho dos métodos em termos de porcentagem de sucesso

Em geral, o desempenho dos problemas com 4 e 7 estágios é semelhante, podendo indicar que o número de estágios não afeta o desempenho de um método.

O método 21, em que o procedimento não utiliza ordenação inicial e observa a regra de alocação SCT, de maneira geral forneceu os melhores resultados, ou seja:

- ✓ Para todas as relações $O(p_i)/O(s_{ij})$ nos problemas de 10 a 40 tarefas, com exceção dos problemas de 10 tarefas para as relações II e VI;
- ✓ Para as relações II, III e VI, nos demais problemas de 50 a 120 tarefas, com exceção dos problemas de 110 tarefas para a relação II;

O gráfico da Figura 2 mostra que, com o aumento do número de tarefas, o método 21 melhora o desempenho nas relações III e VI, piora nas relações I, IV e V, e apresenta certa variação na amplitude do desempenho na relação II (de 1 a 25%). Quanto maior o número de tarefas, maior é variação do desempenho do método para as relações $O(_{pi})/O(s_{ij})$.

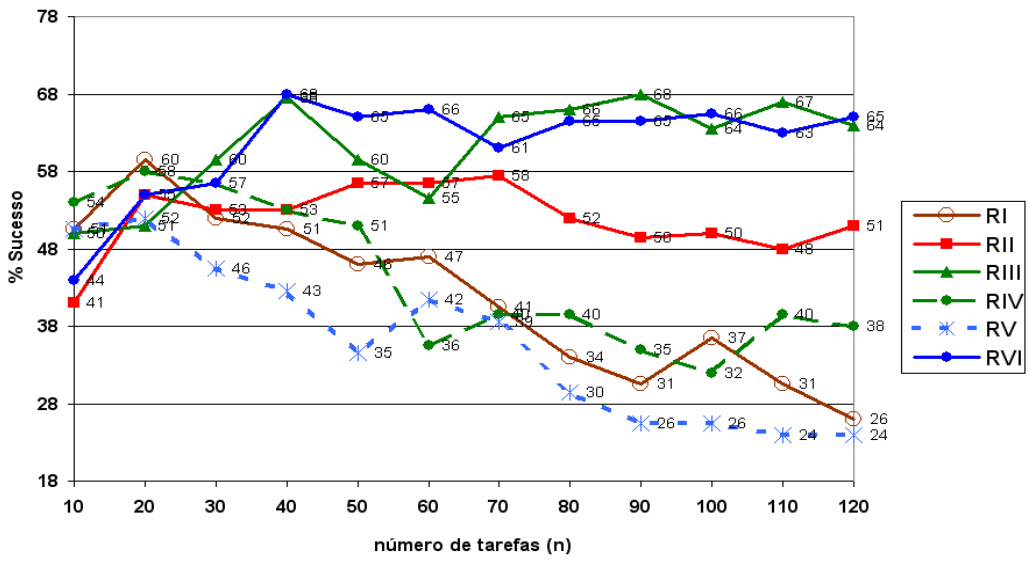

Figura 2 – Comparação da porcentagem de sucesso do método 21 entre as relações $O(p_i)/O(s_{ij})$ juntando resultados de problemas com 4 e 7 estágios

5 Considerações finais

Na literatura, encontram-se poucas pesquisas sobre o desenvolvimento de métodos heurísticos para a programação de *flow shop* híbrico com *setup* dependente da seqüência de processamento das tarefas. Neste trabalho, foram propostos quatro métodos heurísticos construtivos para solução de problemas no ambiente citado. Foi apresentado um estudo sobre a influência da relação entre as ordens de grandeza dos tempos de processamento das tarefas e dos de *setup* das máquinas com base no melhor entre os quatro métodos propostos. Existe uma variabilidade relevante no desempenho dos métodos dependendo da relação $O(p_i)/O(s_{ij})$, com o aumento do número de tarefas, conforme mostra a Tabela 2.

Para o desenvolvimento de futuros trabalhos, sugere-se um estudo "estrutural" do ambiente de produção *flow shop* híbrido com tempos de preparação das máquinas separados dos tempos de processamento das tarefas (dependentes e/ou independentes da seqüência de execução), buscando o desenvolvimento de novos métodos heurísticos eventualmente com melhores desempenhos do que os propostos neste trabalho.

Referências

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

FUCHIGAMI, H.Y. (2005). *Métodos heurísticos construtivos para o problema de programação da produção em sistemas* flow shop *híbridos com tempos de preparação das máquinas assimétricos e dependentes da seqüência*. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos. 2005.

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

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*. Vol.86, n.2, p.133-143, nov.

RAJENDRAN, C. & ZIEGLER, H. (1997) - 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*. Vol.33, n.1-2, p.281-284, oct.

RÍOS-MERCADO, R.Z. & BARD, J.F. (1998) - Heuristics for the flow line problem with setup costs. *European Journal of Operational Research*. Vol.110, p.76-98.

_____. (1999) - An enhanced TSP-based heuristic for makespan minimization in a flow shop with setup times. *Journal of Heuristics*. Vol.5, p.53-70.

SIMONS JR., J.V. (1992) - Heuristics in flow shop scheduling with sequence dependent setup times. *Omega – The International Journal of Management Science*. Vol.20, n2, p.215-225.

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

A pesquisa relatada neste artigo teve o apoio da CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior e do CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico.