COMBINANDO PROCEDIMENTOS NA HEURÍSTICA NEH PARA O PROBLEMA FLOW SHOP COM TEMPOS DE SETUP INDEPENDENTES DA SEQUÊNCIA

Clarissa Tararam de Laurentys, clarissa.laurentys@usp.br[1]
Marcelo Seido Nagano, drnagano@usp.br[2]

[1]Universidade de São Paulo, Av. Trab. São Carlense, 400 - Parque Arnold Schimidt, São Carlos - SP, 13566-590l
[2]Universidade de São Paulo, Av. Trab. São Carlense, 400 - Parque Arnold Schimidt, São Carlos - SP, 13566-590l

Resumo. *Neste artigo, apresenta-se o desenvolvimento e a avaliação de métodos de solução para o problema flow shop com tempos de setup independentes da sequência de produção. O critério de desempenho adotado foi a duração total da programação, também chamada de makespan. A heurística NEH foi analisada e, para reduzir a complexidade computacional, adaptou-se o método da Aceleração de Taillard, considerando os tempos de setup independentes à sequência. No total, foram realizados experimentos computacionais para 48 métodos resultantes da combinação de 12 regras de prioridade utilizadas no primeiro passo do NEH com quatros estratégias de desempate para o quarto passo. O desempenho desses métodos foi avaliado de acordo com seus valores de desvio percentual médio relativo (ARPD) e de tempo computacional percentual médio relativo (ARPT). Os resultados experimentais mostram que o uso de diferentes estratégias de desempate apresenta impactos significantes nas soluções obtidas, sendo que as estratégias FF e RTC apresentaram os melhores valores de ARPD, porém a FF se destacou por apresentar menor tempo de CPU. Já as regras de prioridade provocaram baixos impactos nos resultados, sendo que eles variam de acordo com a estratégia de desempate utilizada.*

Palavras chave*: Flow shop. Setup independente. Scheduling. Makespan.*

Abstract. *This article presents the development and evaluation of solution methods for the flow shop problem with setup time independent of the production sequence. The performance criterion adopted was the maximum of job completion time, also called makespan. The NEH heuristic was analyzed and, to reduce the computational complexity, the Taillard Acceleration method was adapted, considering the setup time independent of the sequence. In total, computational experiments were performed for 48 methods resulting from the combination of 12 priority rules used in the first step of the NEH with four tie-breaking strategies for the fourth step. The performance of these heuristics was evaluated according to their average relative percentage deviation (ARPD) and average relative percentage computational time (ARPT) values. The experimental results show that the use of different tie-breaking strategies has significant impacts on the solutions obtained, and the FF and RTC strategies presented the best ARPD values, but the FF stood out for having lower CPU time. On the other hand, the priority rules had low impacts on the results that vary according to the tie-breaking strategy used.*

Keywords: *flow shop, independent setup, scheduling, makespan.*

1. INTRODUÇÃO

Um problema de programação da produção (*scheduling*) consiste em encontrar uma sequência de tarefas (bens e serviços) que serão processadas em um conjunto de máquinas (equipamentos, colaboradores, máquinas propriamente ditas) com o objetivo de minimizar uma determinada função-objetivo (Baker, 1974; Graham, *et al*., 1979). Um dos ambientes de produção mais estudados desde a década de 50 (Johnson, 1954) é o denominado *Flow Shop* Permutacional, PFSP, (Maccarthy e Liu, 1993; Reza Hejazi e Saghafian, 2005). Esse problema consiste em sequenciar *n* tarefas que devem ser processadas em *m* máquinas dispostas em série, passando pela mesma ordem de processamento e, em cada máquina, obedecendo a regra Primeira que Entra, Primeira que Sai (PEPS).

Uma situação mais realística para o problema de *flow shop* é a consideração do tempo de setup. O tempo de setup ou de preparação de máquina compreende as medidas necessárias para se preparar a máquina para processar determinada tarefa. Isso inclui obter, retornar e ajustar ferramentas, posicionar materiais entre as máquinas, limpeza e inspeção de materiais. O tempo de setup é dependente da sequência se sua duração depende das famílias dos lotes atual e do

imediatamente anterior, e é independente da sequência se sua duração depende apenas da família do lote atual a ser processado (Allahverdi, et al., 2008).

A incorporação do tempo de setup em problemas de programação da produção tem sido estudada desde a década de 60. Na prática, as considerações de setup independente da sequência impactam diretamente na melhor utilização de recursos. Os benefícios da redução do tempo de setup incluem a redução de custos, aumento da velocidade da produção, redução dos *lead* times, maior agilidade no processo de troca de ferramentas e na entrega do produto ao cliente, aumentando sua satisfação (Allahverdi, et al., 2008). Assim, ele pode ser objeto de estudo de vários casos, como montagem e desmontagem de peças ou ferramentas na máquina, limpeza, evacuação de meios de produção, entre outros, o que permite que esses processos sejam realizados por robôs ou manipuladores independentes das tarefas processadas pelas máquinas.

1.1. Objetivo

Neste trabalho analisa-se o problema de *flow shop* permutacional com tempos de setup independentes da sequência (*Permutational Flow Shop Problem with Sequence-Independent Setup Times*, PFSP-SIST), no qual é permitida a existência de estoques intermediários (*buffers*) entre máquinas. O objetivo do trabalho é a análise e proposição de uma nova heurística, obtida por meio de adaptações do primeiro e do quarto passos da heurística NEH, que minimize a duração total da programação (*makespan*).

1.2. Definição do problema

Segundo Belabid, *et al.*, (2019), no problema PFSP-SIST um conjunto de tarefas $J = \{J_1, J_2, \dots, J_n\}$, que constitui um lote de produção, é lançado no processo de produção que contém um conjunto de máquinas em série $M = \{M_1, M_2, \dots, M_m\}$. Todas as tarefas são compostas de m operações, uma em cada máquina. Cada operação da tarefa J_j começa na primeira máquina e termina seu ciclo na última máquina. Considera-se que $p_{i,j}$ é o tempo de processamento e que $s_{i,j}$ é o tempo de setup da tarefa J_j na máquina M_i. O objetivo é encontrar a sequência ótima entre as *n*! possibilidades que minimize o *makespan*.

O cálculo do *makespan* pode ser realizado da maneira tradicional, na qual a complexidade computacional total é de $O(n^3m)$. No entanto, é possível reduzi-la para $O(n^2m)$ utilizando o procedimento da Aceleração de Taillard (Taillard, 1990), que se se baseia no cálculo de três parâmetros, Eq. (1), Eq. (2) e Eq. (3), a fim de obter o *makespan* parcial M_j quando a tarefa k é adicionada na posição j, Eq. (4). Dessa forma, para os experimentos computacionais deste trabalho, utilizou-se este procedimento, adaptando-o para considerar os tempos de setup independentes da sequência.

$$e_{i,j} = max\left(e_{i,j-1}+s_{i,j}\,,\, e_{i-1,j}\right) + p_{i,j},\ e_{0,j}=0,\ e_{i,0}=0\,,\ i = 1, \dots, m,\ j = 1, \dots,\ k\text{-}1 \quad (1)$$

$$q_{i,j} = max\left(q_{i,j+1}+s_{i,j+1}\,,\, q_{i+1,j}\right) + p_{i,j},\ q_{i,k}=0,\ q_{m+1,j}=0,\ i = m, \dots, 1,\ j = k\text{-}1, \dots, 1,\ Para\ j = k,\ s_{i,j+1}=0 \quad (2)$$

$$f_{i,j} = max\left(e_{i,j-1}+s_{ik}\,,\, f_{i-1,j}\right) + p_{k,j},\ f_{i,0}=0,\ i = 1, \dots, m,\ j = 1, \dots, k \quad (3)$$

$$M_j = max_i\left(f_{i,j},\, q_{i,j}+s_{ij}\right),\ i = 1, \dots, m,\ j = 1, \dots, k \quad (4)$$

2. METODOLOGIA

Para a elaboração deste artigo, inicialmente realizamos uma revisão bibliográfica, principalmente de artigos de revistas internacionais indexadas na base SCOPUS e Web of Sciences, em torno do tema *flow shop* relacionado à heurística NEH. A segunda etapa se baseou no desenvolvimento dos métodos. Foram propostos 48 métodos, obtidos pela combinação de 12 regras de prioridade do primeiro passo do NEH com quatros estratégias de desempate do quarto passo. Em seguida, realizou-se experimentos computacionais para todos os métodos, utilizando a base de dados proposta por Ruiz e Allahverdi (2007). A implementação foi efetuada em Python em um PC com CPU Intel Core i5-5200U 2.20 GHz, 8 GB de RAM e operando sob o sistema operacional Windows 10. Em seguida, analisou-se os resultados obtidos a partir dos critérios de desempenho ARPD (*Average Relative Percentage Deviation*) e ARPT (*Average Relative Percentage computational Time*), desempenhos heurísticos e de eficiência computacional, respectivamente. Ao final, foram descritas as principais conclusões identificadas.

Este artigo apresenta, na Seção 3, uma revisão da literatura relacionada ao problema estudado, com foco no primeiro e no quarto passos da heurística NEH e, na Seção 4, a descrição dos 48 métodos avaliados. Os resultados das experimentações computacionais realizadas estão organizados na Seção 5 e a conclusão está exposta na Seção 6.

3. REVISÃO BIBLIOGRÁFICA

O *flow shop* pode ser visto em diversos ambientes produtivos tais como na indústria química, produção de peças aeronáuticas, em tratamento de dejetos (Martinez, *et al.*, 2006), na produção de cidra (Trabelsi, *et al.*, 2012), de placas de circuitos eletrônicos (Gong, *et al.*, 2010), de ferro e aço (Hall e Sriskandarajah, 1996), de blocos de concreto (Grabowski e Pempera, 2000), em operações de terraplanagem (Yu e Seif, 2016) e também no setor de serviços (Hall e Sriskandarajah, 1996).

Para um pequeno número de tarefas e máquinas, métodos exatos, que geram a melhor solução para o problema, têm sido utilizados com sucesso. Entretanto, dependendo do ambiente de produção, à medida que restrições adicionais e o número de tarefas e de máquinas aumentam, torna-se mais complexa a tomada de decisão por parte do programador de produção. Por exemplo, um problema *flow shop* permutacional com *n* tarefas apresenta um total de *n*! soluções. Porém, para m > 2 máquinas, como é o caso da base de dados analisada para a confecção desse projeto, o problema *flow shop* foi provado ser da classe de problemas NP-Difícil (Graham, *et al.*, 1979; Logendran e Sriskandarajah, 1993; Ventura e Yoon, 2013).

Várias heurísticas foram desenvolvidas a fim de minimizar o *makespan* em problemas de permutação flowshop, por exemplo, heurísticas de Page, Palmer, Campbell et al., Gupta, Dannenbring, Nawaz et al. (denotado por NEH) e Hundal et al., e mais recentemente, por Koulamas, Li et al. e Kalczynski et al.. Entre as heurísticas anteriores, a NEH é considerada a melhor (Dong et al., 2007), sendo este um fator relevante para sua análise no contexto PFSP-SIST.

3.1. Heurística NEH

Segundo Nawaz, *et al.* (1982), a heurística NEH é apresentada da seguinte forma:

1. Para cada tarefa *j*, calcule: $T_j = \sum_{i=1}^{m} t_{i,j}$, sendo $t_{i,j}$ o tempo de processamento da tarefa *j* na máquina i;
2. Ordene as tarefas em ordem decrescente de $T_{j.;}$
3. Selecione as duas primeiras tarefas da lista do Passo 2 e as sequencie a fim de minimizar o *makespan* parcial como se a sequência apresentasse apenas essas duas tarefas, use *i* = 3;
4. Posicione a *k*-ésima tarefa da sequência do Passo 2, entre os *i* possíveis, naquele que minimiza o *makespan* parcial sem alterar as posições relativas das tarefas já designadas. O número de repetições desse passo é igual a *k*;
5. Se *n* = 1 pare ou use *k* = *k* + 1 e volte ao Passo 4.

3.1.1. Primeiro passo do NEH

O primeiro passo da heurística NEH consiste na definição de uma regra de prioridade (*priority rule* – PR) que será utilizada no passo seguinte para ordenar as tarefas. Originalmente, como demonstrado na sessão anterior, a regra utiliza o somatório dos tempos de processamentos das tarefas nas máquinas, porém, diversos autores criaram novas métricas de ordenação de tarefas que apresentaram resultados significantes quando comparados à regra original.

Dong, *et al.* (2007) analisaram o comportamento de três regras de priorização: $PR_{Avg,}$ PR_{Dev} e PR_{AvgDev}, que ordenam as tarefas de acordo com o tempo médio de processamento da tarefa j, AVG_j, (o mesmo que é utilizado na NEH original), com o desvio padrão dos tempos de processamento, STD_j, e com a soma entre AVG_j e STD_j, respectivamente.

Já Liu, *et al.* (2017) propuseram outra regra de prioridade, PR_{SKE}, na qual as tarefas são ordenadas em maneira não crescente a partir do somatório entre AVG_j, STD_j e o valor absoluto da distorção da tarefa j, $abs(SKE_j)$, que é uma medida de assimetria de uma distribuição de probabilidade. Com esse novo termo foi possível evitar empates que eram comuns no momento de ordenação, uma vez que tarefas diferentes podem compartilhar a mesma média e desvio padrão.

3.1.2. Quarto passo do NEH

O quarto passo da heurística NEH consiste no sequenciamento das tarefas de forma a minimizar o *makespan* da sequência parcial. Porém, a inserção de uma nova tarefa em diferentes posições pode gerar o mesmo *makespan* parcial**.** Dessa forma, várias estratégias de desempates (*tiebreaking strategy* – TB) foram desenvolvidas e comparadas com a original (TB_{NEH}), na qual, em caso de empate, a primeira sequência parcial, ou seja, o primeiro local de inserção, é mantido.

Dong, *et al*. (2007) propuseram uma estratégia de desempate baseada na escolha do local com maior probabilidade de equilibrar a utilização de cada máquina, TB_D. Para isso, as medidas $E_{\pi(x)}$ e $D_{\pi(x)}$ são calculadas e, em caso de empates, a sequência parcial que apresentar o mínimo D é escolhida. Já Fernandez-Viagas e Framinan (2014) propuseram uma estratégia baseada na minimização do tempo ocioso das máquinas, TB_{FF}. Essa proposta desempata as sequências parciais escolhendo aquela que apresenta o mínimo tempo ocioso, $it(l)$, quando a tarefa é inserida na posição l.

Ribas, *et al*. (2010) propuseram outra estratégia de desempate, TB_{RTC}, baseada em dois métodos de desempate. O primeiro objetiva minimizar o tempo total de ociosidade das máquinas, porém, em caso de empates, utiliza-se o segundo, proposto por Kalczynski e Kamburowski (2008), no qual calcula-se as medidas a_j e b_j, assim, a posição escolhida para

tarefa inserta é aquela mais próxima da primeira posição da sequência parcial se $a_j \leq b_j$, caso contrário é aquela mais próxima da última posição.

4. MÉTODOS PROPOSTOS

Foram propostos 48 métodos, obtidos pela combinação de 12 regras de prioridade, Tab. 1, do primeiro passo do NEH com quatros estratégias de desempate do quarto passo. Das 12 regras de prioridade, três são adaptações daquelas apresentadas na Seção 3.1.1., PR_{Avg}, PR_{AvgDev} e PR_{SKE}, e as outra nove foram desenvolvidas a partir da combinação de diferentes métricas relacionadas aos tempos de processamento e de setup.

Tabela 1. Regras de prioridade adaptadas e desenvolvidas (Laurentys e Nagano, 2022)

Regra de Prioridade	Cálculo utilizado
PR_{Avgs}	$AVG_{proc} + AVG_{setup}$
$PR_{AvgDevs}$	$AvgDev_{proc} + AvgDev_{setup}$
$PR_{skeAvgDevs}$	$AvgDev_{proc} + AvgDev_{setup} + SKE_{proc} + SKE_{setup}$
PR_{AvgDev_avg}	$AvgDev_{proc} + Avg_{setup}$
PR_{avg_AvgDev}	$Avg_{proc} + AvgDev_{setup}$
PR_{avg_max}	$Avg_{proc} + max_{setup}$
PR_{AvgDev_max}	$AvgDev_{proc} + max_{setup}$
$PR_{skeavgs}$	$Avg_{proc} + Avg_{setup} + SKE_{proc} + SKE_{setup}$
$PR_{ske\_AvgDev\_avg}$	$AvgDev_{proc} + Avg_{setup} + SKE_{proc} + SKE_{setup}$
$PR_{ske\_avg\_AvgDev}$	$Avg_{proc} + AvgDev_{setup} + SKE_{proc} + SKE_{setup}$
$PR_{ske\_avg\_max}$	$Avg_{proc} + max_{setup} + SKE_{proc} + SKE_{setup}$
$PR_{ske\_AvgDev\_max}$	$AvgDev_{proc} + max_{setup} + SKE_{proc} + SKE_{setup}$

Sendo o cálculo de AVG e $AvgDev$ realizado de acordo com as equações de Dong, *et al.* (2007) e de SKE de acordo com Liu, *et al.* (2017). O cálculo de max_{prep} está relacionado ao valor máximo do tempo de setup da tarefa com relação às máquinas ($max_{1 \leq i \leq m}\{s_{i,j}\}$).

As quatro estratégias de desempate são: a estratégia original da heurística NEH, TB_{NEH}, e adaptações daquelas apresentadas na Seção 3.1.2., TB_D, TB_{FF} e TB_{RTC}. O passo a passo e as medidas utilizadas destas estratégias originais são mantidos, apenas os cálculos dos parâmetros $E_{\pi(x)}$, $D_{\pi(x)}$, $it(l)$, $f'_{i,l}$, IT_i, $f_{i,j}$, a_j e b_j foram modificados, vide Tab. 2.

Tabela 2. Parâmetros modificados das estratégias de desempate TB_D, TB_{FF} e TB_{RTC} (Laurentys e Nagano, 2022)

Estratégia de desempate	Parâmetros modificados
TB_D	$E_{\pi(x)} = \frac{1}{m} * \sum_{i=1}^{m} \frac{p_{i,\pi(x)} + s_{i,\pi(x)}}{S_{i,\pi(x+1)} - C_{i,\pi(x-1)}}$, $x = 1, \ldots, n$ $D_{\pi(x)} = \sum_{i=1}^{m} \left(\frac{p_{i,\pi(x)} + s_{i,\pi(x)}}{S_{i,\pi(x+1)} - C_{i,\pi(x-1)}} - E_{\pi(x)}\right)^2$, $x = 1, \ldots, n$
TB_{FF}	$it(l) = \sum_{i=1}^{m} (f_{i,l} - e_{i,l} + p_{i,l} + s_{i,l} + max\{f'_{i,l} - f_{i,l}, 0\})$, $para\ l = k\ e\ it(l) = \sum_{i=1}^{m} (f_{i,l} - e_{i,l-1})$ $f'_{i,l} = max\{f_{i,l} + s_{i,l}, f'_{i-1,l}\} + p_{i,l}$, $i = 1, \ldots, m$, $f'_{0,l} = 0$

TB_{RTC}	$IT_i = f_{i,n} - e_{i,1} - \sum_{j=1}^{n} p_{i,j} - \sum_{j=1}^{n} s_{i,j}$ $f_{i,j} = e_{i,j} + p_{i,j} + s_{i,j},\ i = 1, \ldots, m,\ j = 1, \ldots, n,\ f_{0,j} = 0,\ f_{i,0} = 0$ $a_j = \sum_{i=1}^{m} \left((m-1) * \frac{m-2}{2} + m - i \right) * (p_{i,j} + s_{i,j})$ $b_j = \sum_{i=1}^{m} \left((m-1) * \frac{m-2}{2} + i - 1 \right) * (p_{i,j} + s_{i,j})$

5. RESULTADOS

Os desempenhos das 48 heurísticas foram avaliados pelos critérios ARPD e ARPT. O ARPD remete à média do desvio percentual relativo, chamado de RPD, calculado de acordo com Belabid, *et al.*, (2019). Pode-se observar que quanto menor o valor do RPD, melhor o desempenho heurístico, pois mais próximas estarão suas soluções do melhor resultado encontrado entre todos os métodos comparados (Rossi e Nagano, 2019). Já o ARPT é a média do tempo computacional percentual relativo, chamado de RPT, calculado de acordo com a versão aprimorada de Fernandez-Viagas, *et al.* (2017) e Fernandez-Viagas e Framinan (2017). O ARPT mede o esforço computacional dos métodos (Fernandez-Viagas e Framinan, 2017), assim, quanto menor o seu valor, menor é o esforço computacional necessário para a execução do método.

Os resultados obtidos a partir dos métodos avaliados estão resumidos na Tab. 3, na qual os melhores e os piores resultados em cada coluna estão em negrito e em vermelho, respectivamente. A partir dela pode-se observar que, os melhores valores de ARPD nas distribuições 50% e 100% foram obtidos pelos métodos compostos pela estratégia de desempate RTC e, na distribuição 150% e na média entre as três distribuições, pelos métodos compostos pela estratégia de desempate FF. Ademais, em todos os casos, os métodos compostos pela estratégia de desempate NEH apresentaram os piores valores de ARPD e os melhores de média de tempo de CPU e de ARPT, sendo que os métodos compostos pela estratégia de desempate D apresentaram os piores resultados para estas duas últimas métricas.

Tabela 3. Resumo dos resultados obtidos (Laurentys e Nagano 2021)

	ARPD					
	50%	100%	150%	Média ARPD	Média de tempo de CPU (ms)	ARPT
AvgDevsNEH	0,434	0,394	0,424	0,417	2996	0,475
AvgDevsD	0,371	0,388	0,378	0,379	14642	1,760
AvgDevsFF	0,265	0,263	0,262	0,263	3134	0,540
AvgDevsRTC	0,265	0,269	0,270	0,268	10244	1,254
AvgDev_avgNEH	0,421	0,416	0,398	0,412	3001	0,478
AvgDev_avgD	0,369	0,368	0,357	0,365	14417	1,742
AvgDev_avg FF	0,271	0,262	0,263	0,266	3132	0,540
AvgDev_avgRTC	**0,249**	0,263	0,285	0,266	10360	1,246
avg_AvgDevNEH	0,430	0,401	0,414	0,415	2998	0,478
avg_AvgDevD	0,389	0,374	0,363	0,375	14070	1,732
avg_AvgDevFF	0,253	0,267	0,271	0,264	3131	0,540
avg_AvgDevRTC	0,267	0,265	0,271	0,268	10143	1,229
AvgDev_maxNEH	0,438	0,405	0,428	0,424	3000	0,478
AvgDev_maxD	0,363	0,380	0,375	0,373	14745	**1,779**
AvgDev_maxFF	0,257	0,271	0,251	0,260	3134	0,545
AvgDev_maxRTC	0,253	0,254	0,283	0,263	10476	1,264
avgNEH	0,438	0,413	0,394	0,415	2997	0,477
avgsD	0,383	0,381	0,345	0,370	14019	1,696
avgsFF	0,282	0,274	0,246	0,267	3129	0,538
avgsRTC	0,262	0,266	0,273	0,267	10042	1,206
avg_maxNEH	0,442	0,391	0,416	0,417	**2994**	0,477

avg_maxD	0,369	0,379	0,377	0,375	14538	1,743
avg_maxFF	0,253	0,275	0,288	0,272	3134	0,541
avg_maxRTC	0,256	**0,242**	0,272	0,257	10207	1,236
ske_AvgDevsNEH	0,446	0,406	**0,432**	0,428	2998	0,475
ske_AvgDevsD	0,381	0,355	0,368	0,368	14605	1,765
ske_AvgDevsFF	0,266	0,250	0,244	**0,253**	3136	0,546
ske_AvgDevsRTC	0,261	0,261	0,276	0,266	10275	1,251
ske_AvgDev_avgNEH	0,447	**0,435**	0,414	0,432	2997	0,477
ske_AvgDev_avgD	0,377	0,375	0,375	0,376	14724	1,746
ske_AvgDev_avgFF	0,256	0,273	**0,242**	0,257	3131	0,543
ske_AvgDev_avgRTC	0,263	0,253	0,275	0,264	10278	1,243
ske_avg_AvgDevNEH	**0,462**	0,416	0,425	**0,435**	2996	**0,472**
ske_avg_AvgDevD	0,399	0,376	0,369	0,381	14115	1,729
ske_avg_AvgDevFF	0,264	0,284	0,258	0,269	3131	0,540
ske_avg_AvgDevRTC	0,281	0,260	0,272	0,271	10193	1,226
ske_AvgDev_maxNEH	0,441	0,421	0,426	0,430	2999	0,479
ske_AvgDev_maxD	0,382	0,378	0,375	0,378	**14841**	1,778
ske_AvgDev_maxFF	0,254	0,271	0,276	0,267	3134	0,540
ske_AvgDev_maxRTC	0,260	0,263	0,268	0,264	10417	1,261
ske_avgsNEH	0,444	0,407	0,409	0,420	2999	0,477
ske_avgsD	0,379	0,376	0,374	0,376	14348	1,710
ske_avgsFF	0,277	0,254	0,263	0,265	3132	0,540
ske_avgsRTC	0,257	0,275	0,257	0,263	10111	1,213
ske_avg_maxNEH	0,442	0,419	0,409	0,424	2997	0,474
ske_avg_maxD	0,384	0,376	0,368	0,376	14153	1,744
ske_avg_maxFF	0,259	0,282	0,267	0,270	3135	0,538
ske_avg_maxRTC	0,263	0,243	0,263	0,257	10220	1,240

Dessa forma, entende-se que o impacto nos resultados obtidos é resultante, primordialmente, da escolha da estratégia de desempate. A Figura 1 torna os resultados obtidos mais visíveis, na qual cada círculo representa um dos 48 métodos. Porém, é evidente a formação de quatro zonas, cada uma composta pela combinação das 12 regras de prioridade com uma das estratégias de desempate.

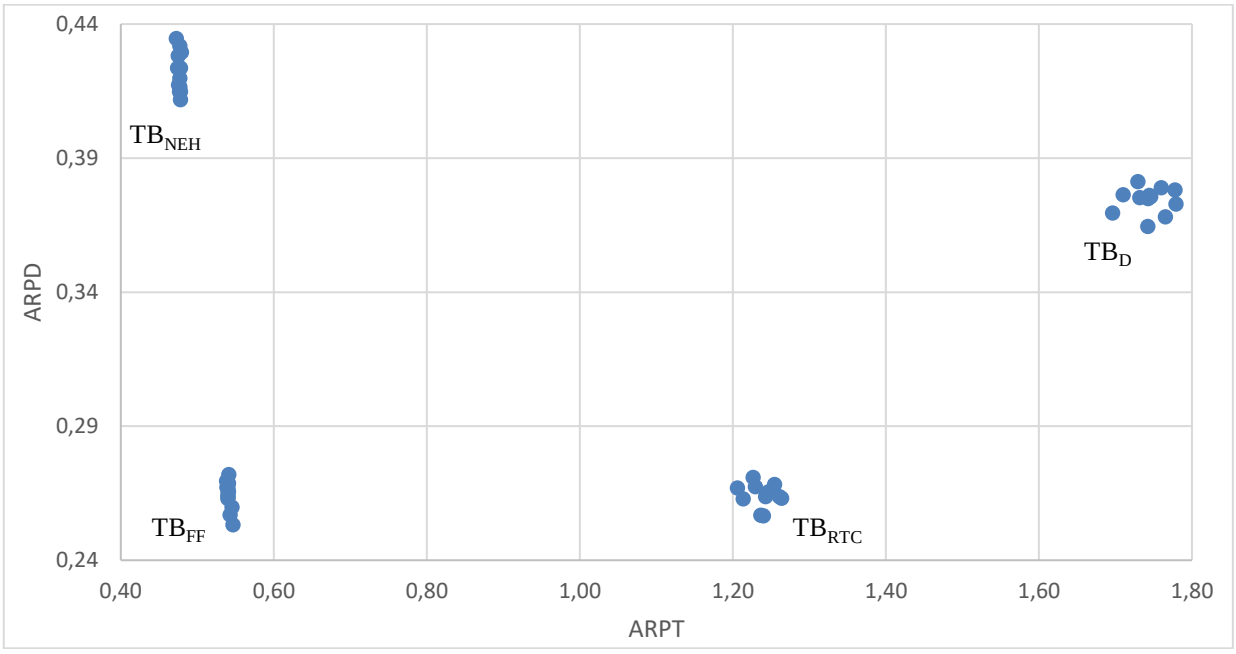

Figura 1. Valores de ARPD e ARPT em relação ao número de tarefas para os métodos comparados (Laurentys e Nagano 2021)

A primeira zona, composta pela estratégia NEH, apresenta os menores valores de ARPT e os maiores valores de ARPD; a segunda, composta pela estratégia FF, apresenta valores baixos de ARPT e os menores valores de ARPD, semelhantes aos encontrados na terceira zona, composta pela estratégia RTC, que apresenta valores medianos de ARPT; e, por fim, a quarta zona, composta pela estratégia D, apresenta os maiores valores de ARPT e valores medianos de ARPD. Porém, não foi possível identificar nenhum padrão entre a utilização das regras de prioridades e o desempenho do método, pois uma mesma regra apresenta impactos diferentes quando combinada com diferentes estratégias de desempate. Estes impactos podem ser observados nas Fig. 2, 3-5.

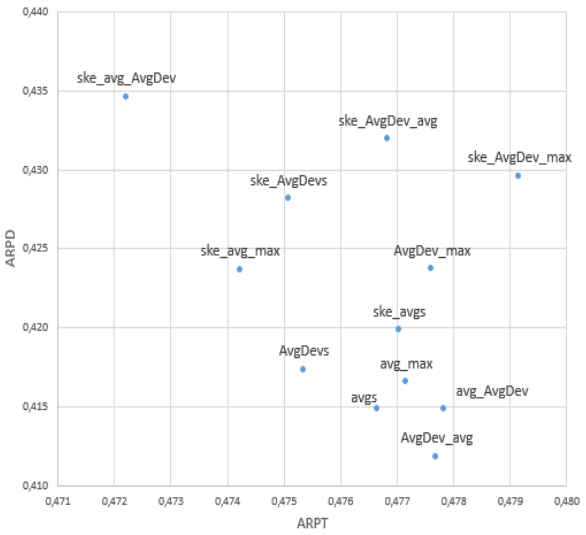

Figura 2. Valores de ARPD e ARPT para os métodos que utilizam a estratégia de desempate NEH (Laurentys e Nagano 2021)

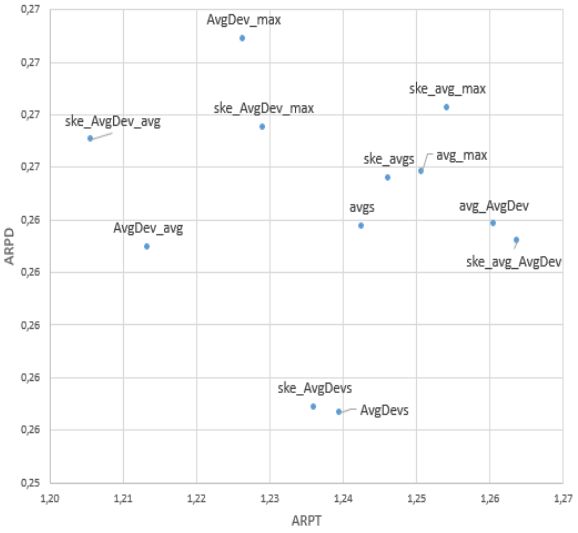

Figura 4. Valores de ARPD e ARPT para os métodos que utilizam a estratégia de desempate RTC (Laurentys e Nagano 2021)

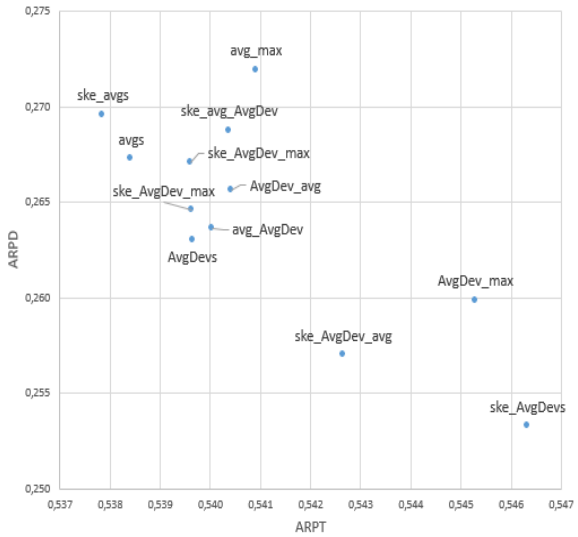

Figura 3. Valores de ARPD e ARPT para os métodos que utilizam a estratégia de desempate FF (Laurentys e Nagano 2021)

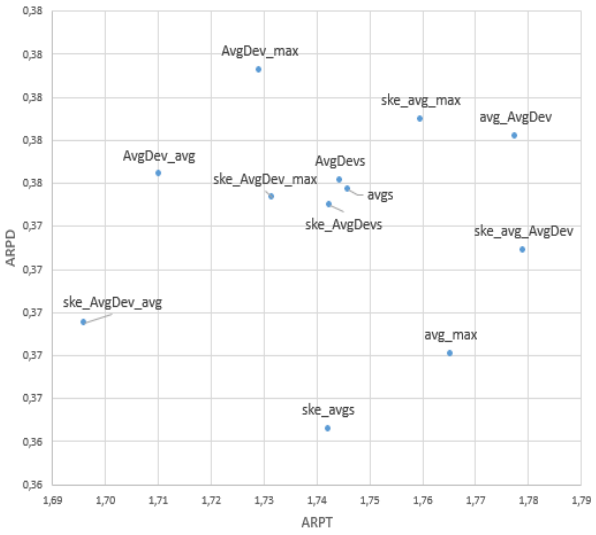

Figura 5. Valores de ARPD e ARPT para os métodos que utilizam a estratégia de desempate D (Laurentys e Nagano 2021)

6. CONCLUSÕES

Neste artigo foi abordado o problema de programação *flow shop* permutacional com a restrição de tempos de setup independentes à sequência (PFSP-SIST) com o objetivo de encontrar a sequência entre as n! possibilidades que minimize o tempo máximo de conclusão, chamado de *makespan*. A heurística em análise é a NEH, para a qual foram propostos 48 métodos a partir da combinação de 12 regras de prioridade, utilizadas no primeiro passo da NEH, com 4 estratégias de desempate, utilizadas no quarto passo.

A avaliação dos 48 métodos resultou em quatro grandes conclusões: i) o uso de diferentes estratégias de desempate apresenta grande impacto nos resultados finais; ii) o impacto da regras de prioridade nos resultados finais variam de acordo com a estratégia de desempate utilizada; iii) todas as estratégias de desempate apresentaram melhores valores de ARPD quando comparadas à estratégia original, NEH; iv) as estratégias FF e RTC apresentaram os melhores valores de ARPD, porém a FF se destacou por apresentar menor tempo de CPU.

Dessa forma, evidencia-se a superioridade dos métodos que utilizam a estratégia de desempate FF, sendo que, entre elas, a variação no uso das regras de prioridade gera pequenas diferenças nos valores de ARPD e ARPT. Outra contribuição deste trabalho é a adaptação do algoritmo da aceleração de Taillard, que reduz a complexidade computacional de $O(n^3m)$ para $O(n^2m)$, de forma a incluir os tempos de setup independentes à sequência de tarefas.

Em futuras pesquisas seria interessante a análise de diferentes algoritmos normalmente utilizados em problemas que consideram apenas o tempo de processamento das tarefas a fim de adaptá-los a restrição de tempos de setup independentes. Além disso, implementar os métodos aqui expostos em diferentes bases de dados, que apresentem diferentes variações no número de tarefas e máquinas, é outra possibilidade a ser explorada.

7. REFERÊNCIAS

Allahverdi, A.; Ng, C. T.; Cheng, T. E.; Kovalyov, M. Y, 2008. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, v. 187, n. 3, p. 985-1032.

Baker, K. R, 1974. Introduction To Scheduling and Sequencing. Nova York: Wiley.

Belabid, J.; Aqil, S.; Karam, A., 2020. Solving Permutation Flow Shop Scheduling Problem with Sequence-Independent Setup Time. *Journal of Applied Mathematics*, v. 2020, Article ID 7132469, p. 11.

Dong, X.; Huang, H.; Chen, P, 2008. An improved NEH-based heuristic for the permutation flowshop problem. *Computers & Operations Research*, v. 35, p. 3962–3968.

Fernandez-Viagas, V.; Framinan, J. M, 2014. On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem. *Computers & Operations Research*, v. 45, p. 60–67.

Grabowski, J.; Pempera, J, 2000. Sequencing of jobs in some production system. *European Journal of Operational Research*, v. 125, n. 3, p. 535–550, set. 2000.

Graham, R. L.; Lawler, E. L.; Lenstra, J. K.; Kan, A. R, 1979. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete mathematics*, v. 5, p. 287-326.

Hall, N. G.; Sriskandarajah, C, 1996. A survey of machine scheduling problems with blocking and no-wait in process. *Operations Research*, v. 44, n. 3, p. 510–525.

Johnson, S. M, 1954. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, v. 1, n. 1, p. 61–68.

Kalczynski, P. J.; Kamburowski, J, 2008. An improved NEH heuristic to minimize makespan in permutation flow shops. *Computers & Operations Research*, v. 35, p. 3001– 3008.

Liu, W.; Jin, Y.; Price, M, 2017. A new improved NEH heuristic for permutation flowshop scheduling problems. *International Journal of Production Economics*, v. 193, p. 21–30.

Logendran, R.; Sriskandarajah, C, 1993. Two-machine group scheduling problem with blocking and anticipatory setups. *European Journal of Operational Research*, v. 69, n. 3, p. 467–481.

Maccarthy, B. L.; Liu, J, 1993. Addressing the gap in scheduling research: a review of optimization and heuristic methods in production scheduling. *International Journal of Production Research*, v. 31, n. 1, p. 59–79.

Martinez, S.; Dauzère-Pérès, S.; Gueret, C.; Mati, Y.; Sauer, N, 2006. Complexity of flowshop scheduling problems with a new blocking constraint. *European Journal of Operational Research*, 169, n. 3, p. 855-864.

Nawaz,M., Ensore Jr., E., Ham, I., 1982. A Heuristic Algorithm for the m-Machine, n-Job Flow-shop Sequencing Problem. *The Int. JI of Mgmt Sci*., v. 11, p. 91-95.

Ribas, I.; Companys, R.; Tort-Martorell, X, 2010. Comparing three-step heuristics for the permutation flow shop problem. *Computers & Operations Research*, v. 37, p. 2062–2070.

Rossi, F. L.; Nagano, M. S, 2019. Heuristics for the mixed no-idle flowshop with sequence-dependent setup times. *Journal of the Operational Research Society*, 72:2, 417-443, DOI: 10.1080/01605682.2019.1671149.

Ruiz, R.; Allahverdi, A, 2007. Some effective heuristics for no-wait flowshops with setup times to minimize total completion time. *Springer Science+Business Media*, p. 143–171.

Taillard, E., 1990. Some efficient heuristic methods for the flow shop sequencing problem. *European Journal of Operational Research*, v.47, p. 65-74.

Trabelsi, W.; Sauvey, C.; Sauer, N, 2012. Heuristics and metaheuristics for mixed blocking constraints flowshop scheduling problems. *Computers and Operations Research*, v. 39, n. 11, p. 2520–2527.

Ventura, J. A.; Yoon, S.-H, 2013. A new genetic algorithm for lot-streaming flow shop scheduling with limited capacity buffers. *Journal of Intel*, v. 24, p. 1185–1196.

Yu, A. J.; Seif, J, 2016. Minimizing tardiness and maintenance costs in flow shop scheduling by a lower-bound-based GA. *Computers and Industrial Engineering*, v. 97, p. 26–40.