Avaliação de métodos heurísticos em sistemas de produção no-wait flow shop

Marcelo Seido Nagano (USP) - drnagano@usp.br
Lucas Yamada Scardoelli (USP) - scarty@terra.com.br
João Vitor Moccellin (USP) - jvmoccel@sc.usp.br

Resumo
Este artigo aborda o problema de programação de operações em um ambiente de produção Flow Shop, tendo como objetivo minimizar o estoque em processamento sem interromper a execução das tarefas. Um novo método heurístico foi proposto e comparado com quatro melhores métodos reportados na literatura para a solução do problema. A análise dos resultados experimentais evidenciou a superioridade do novo método para o conjunto de problemas avaliados.
Palavras-chave: *No-wait flow shop, estoque de processo e métodos heurísticos.*

HEURISTIC EVALUATION OF METHODS OF PRODUCTION SYSTEMS IN NO-WAIT FLOW SHOP

Abstract
This paper deals with the Flow Shop scheduling problem with the objective of minimizing total flow time without interrupting the execution of tasks. A new heuristics is proposed and compared with the four best methods reported in the literature for the solution of the scheduling problem. Analysis of the experimental results shows that the new heuristics provides better solutions to the set of problems evaluated.
Keywords: *No-wait flow shop, in-process inventory and heuristics.*

1. INTRODUÇÃO

O problema tradicional de Programação *Flow Shop* é de produção, em que um conjunto de *n* tarefas deve ser processado, na mesma seqüência, por um conjunto de *m* máquinas. Quando a ordem de processamento em todas as máquinas for a mesma, tem-se o ambiente de produção *Flow Shop* Permutacional, em que o número de possíveis programações para *n* tarefas é *n!*

Uma situação específica desse problema, mas muito freqüente, é a programação da produção em sistema *flow shop* sem interrupção na execução das tarefas, que geralmente ocorre em um ambiente caracterizado pela tecnologia do processo, i.e., quando, por exemplo, uma variação de temperatura pode influenciar a degeneração do material, ou pela falta de capacidade de armazenamento entre as máquinas.

Esse problema também pode ser chamado de *flow shop* sem espera ou *no-wait flow shop* (*NWFS*), conforme apresentado na Figura 1.

A Figura 1 ilustra a programação de um conjunto de *n* tarefas ($J = \{J_1, J_2,...,J_i,...J_n\}$) que devem ser processadas, na mesma seqüência, por um conjunto de *m* máquinas distintas, em que o tempo de processamento da tarefa J_i na máquina *k* é t_{ik} ($i = 1,2,...,n; k = 1,2...,m$).

A principal característica do *NWFS* é a necessidade de que a operação $g + 1$ de determinada tarefa J_i tem que ser processada logo após o término da operação $g (1 \leq g \leq m - 1)$, não permitindo que ocorra tempo de espera no processamento de uma tarefa de uma máquina para a outra (*no-wait*). O único tempo de espera permitido é no início do processamento da tarefa que ocupa a primeira posição na seqüência na primeira máquina.

Devido à sua natureza, o problema *NWFS* é considerado como *NP-hard* para o caso de três ou mais máquinas (CHEN et al., 1996).

Segundo Fink e Voβ (2003) em um problema *NWFS* geralmente se verifica a ocorrência de um *delay* (d_{uv}) na primeira máquina, entre o início da tarefa *u* e o início da tarefa *v*, quando a tarefa *v* é processada diretamente após a tarefa *u*, em que o cálculo do *delay* pode ser efetuado pela seguinte expressão:

$$d_{uv} = t_{u1} + \max_{k}\left(\sum_{p=2}^{k} t_{up} - \sum_{p=1}^{k-1} t_{vp}, 0\right) \quad (1)$$

O objetivo deste trabalho é apresentar um novo método heurístico para o problema de programação de operações em ambientes *NWFS* com critério de minimização do tempo total de fluxo, verificando-se o seu desempenho comparado com os melhores métodos heurísticos existentes na literatura.

Para o problema apresentado, o tempo total de fluxo pode ser obtido conforme a seguinte expressão:

$$F = \sum_{i=2}^{n}(n+1-i)d_{[i-1][i]} + \sum_{i=1}^{n}\sum_{k=1}^{m} t_{ik} \quad (2)$$

Resumidamente, essa expressão fornece a soma dos tempos de processamento das tarefas em todas as máquinas e a soma dos delays entre as tarefas seqüenciadas.

As características essenciais a serem consideradas no novo método proposto serão: equilíbrio entre a qualidade da solução e eficiência computacional e simplicidade e facilidade de implementação.

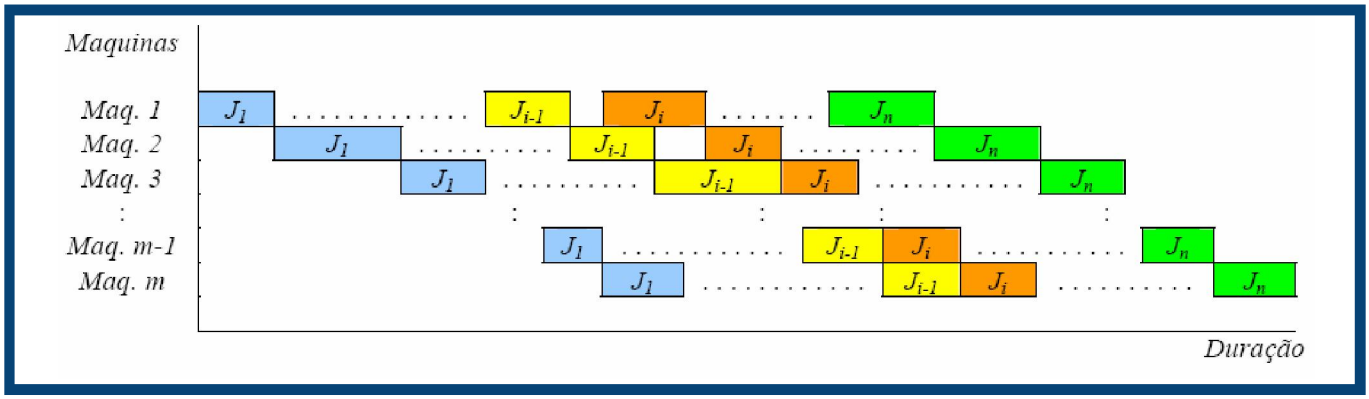

Figura 1 - *No-Wait Flow Shop* com *m* máquinas e *n* tarefas.
Fonte: Elaborada pelos autores.

2. PROBLEMA DE PROGRAMAÇÃO DE OPERAÇÕES *NWFS*

Deman e Baker (1974) foram alguns dos pioneiros a estudar o problema de ***NWFS***, com o critério de minimização do tempo total de fluxo. Em sua pesquisa, eles apresentaram um algoritmo *branch and bound* para estabelecer todas as seqüências parciais, considerando-se a utilização de limitantes inferiores. Os referidos autores concluíram que os resultados alcançados foram satisfatórios e que o problema podia ser solucionado tão rapidamente quanto aqueles tradicionais que utilizavam como critério a minimização do *makespan*.

Rajendran e Chaudhuri (1990) apresentaram dois algoritmos heurísticos construtivos, compostos de duas fases: uma primeira fase de ordenação inicial das tarefas e outra de construção da seqüência final avaliando seqüências parciais, a fim de estabelecer a seqüência final das tarefas. A experimentação computacional demonstrou que as soluções obtidas eram melhores quando comparadas com os métodos anteriormente propostos por Bonney e Gundry (1976) e King e Spachis (1980), com o critério de minimização do *makespan*.

Chen et al. (1996) desenvolveram uma heurística, baseada no algoritmo genético, e, através da sua parametrização, conseguiram melhorar as soluções pelo método construtivo proposto por Rajendran e Chaudhuri (1990).

Bertolissi (1999) apresentou um método heurístico construtivo de apenas uma fase, na qual é definida uma parcela da soma dos tempos de fluxo de duas tarefas adjacentes J_i e J_j, a partir do início de J_i (ou seja, no intervalo de tempo entre o início de J_i e o termino de J_j), sendo em seguida obtida uma ordenação das tarefas. Os experimentos computacionais indicaram que o método proposto não obtém soluções melhores que os métodos de Rajendran e Chaudhuri (1990).

Bertolissi (2000) apresenta um novo método heurístico construtivo composto de duas fases. Na primeira, similar à de Bertolissi (1999), é definida uma ordenação inicial das tarefas de acordo com as parcelas das somas dos tempos de fluxo dos pares tarefas adjacentes J_i e J_j. Na segunda fase, é utilizado o mesmo procedimento de inserção de tarefas de Rajendran e Chaudhuri (1990). Os resultados dos experimentos computacionais permitiram a conclusão de que o método heurístico construtivo proposto obtém soluções melhores quando comparado com os métodos heurísticos construtivos.

Fink e Voâ (2003), utilizando os procedimentos Meta-Heurísticos de Busca Tabu e *Simulated Annealing*, concluíram que a efetividade dos procedimentos meta-heurísticos utilizados na solução do problema depende da qualidade da solução desejada e do tempo computacional disponível. Ou seja, os autores não contemplaram um método como o melhor entre todos, mas afirmaram que, dependendo das características do problema, existe um que pode ser adequado.

Aldowaisan e Allahverdi (2004) propuseram novos métodos heurísticos compostos de três fases. Segundo os resultados dos experimentos, o algoritmo PH1(p) foi o que apresentou os melhores resultados, comparados com os métodos de Rajendran e Chaudhuri (1990) e o método meta-heurístico de Chen et al. (1996). O método proposto é composto pelas seguintes fases: na primeira, é desenvolvida uma ordenação inicial a partir da tarefa que apresenta a menor soma dos tempos de processamento e, em seguida, são ordenadas as tarefas que resultam no menor tempo de total de fluxo na ordenação parcial; na segunda, é obtida uma solução inicial semelhante ao método de inserção proposto por Nawaz et al. (1983), enquanto na última é aplicado o procedimento de melhoria, utilizando a permutação de pares de tarefas.

3. MÉTODO HEURÍSTICO PROPOSTO

O método heurístico proposto neste trabalho possui duas fases: na primeira é realizada uma ordenação inicial das tarefas, utilizando-se a expressão 3; e na segunda busca-se construir a seqüência de solução com a inserção de tarefas nas seqüências parciais e sua melhoria com um método de busca na vizinhança. Para fins de comparação, o referido método é denominado **S&N**.

O método heurístico proposto por Bertolissi (2000) apresenta uma expressão que o próprio autor considerou como a base do seu método proposto. Trata-se de uma parcela da soma dos tempos de fluxo de duas tarefas adjacentes J_i e J_j, a partir do início de J_i (ou seja, no intervalo de tempo entre o início de J_i e o término de J_j) para m máquinas.

Segundo Bertolissi (2000), é possível, dessa forma, calcular as parcelas das somas dos tempos de fluxo para o conjunto de pares de tarefas adjacentes para m máquinas com a seguinte expressão:

$$F_m(J_i, J_j) = 2t_{i1} + \sum_{k=2}^{m} t_{ik} + R_m(J_i, J_j) \quad (3)$$

em que:

$$R_1(J_i, J_j) = t_{j1}$$

$$R_m(J_i, J_j) = t_{jm} + max\left(R_{m-1}(J_i, J_j), \sum_{k=2}^{m} t_{ik} \right)$$

Na expectativa de minimizar o tempo total de fluxo, a expressão 3 possibilita a criação de uma ordenação inicial das tarefas a partir das menores parcelas das somas dos tempos de fluxo de pares de tarefas adjacentes.

A adoção dessa forma de ordenação inicial é baseada no fato de que, para minimização do tempo total de fluxo de determinada seqüência de tarefas, devem-se ordenar as tarefas que apresentam menores tempos de processamentos. Porém, para atender ao caso particular de *NWFS*, é necessário considerar também o *delay* entre o início de determinada tarefa e o início da tarefa subseqüente.

1ª Fase - Ordenação inicial das tarefas

Assim, seja $J = \{J_i, J_2, \ldots J_n\}$ um conjunto de n tarefas que devem ser processadas e S, o conjunto das tarefas programadas.

Passo 1

$J = \{J_i, J_2, \ldots, J_i, \ldots J_n\}$

S = Ø

Selecione o menor elemento $Fm\ (J_i, J_j)$

$S = \{J_i, J_j\}$:

$J \leftarrow J - \{J_i, J_j\}$

$u \leftarrow J_{[2]}$

$k = 3$

Passo 2

Selecione o menor elemento $F_m(J_u, J_v)$, tal que $J_v \in J$

$S \leftarrow S \cup \{J_v\}$

$J \leftarrow J - \{J_v\}$

$u \leftarrow J_{[k]}$

$k \leftarrow k + 1$

Se $J \neq$ Ø, volte para o **Passo 2**. Caso contrário (ordenação inicial das tarefas concluída), vá para o **Passo 3**.

2ª Fase - Construção da seqüência final

A construção da seqüência final é uma modificação do algoritmo proposto por Framinan e Leisten (2003) para o problema NWFS.

A modificação ocorre na segunda fase do algoritmo, em que para cada iteração é aplicado um procedimento de melhoria na seqüência parcial, utilizando-se a vizinhança de inserção de tarefas.

Essa estrutura possibilita avaliar o maior número de seqüências parciais, em comparação com o método proposto por Framinan e Leisten (2003), não inviabilizando o tempo de computação.

A segunda parte do algoritmo proposto é definida pelos seguintes passos:

Passo 3

Selecione as duas primeiras tarefas da ordenação inicial

Passo 4

Para $L = 3$ a n

- Selecione a tarefa que ocupa a L – *ésima* posição na ordenação inicial .
- Examine as L possibilidades de inserir a tarefa na seqüência parcial até então obtida, adotando-se aquela que leva a menor soma dos tempos de fluxo.
- Considerando toda a Vizinhança de Inserção da seqüência parcial com L tarefas, constituída de S' seqüências, determine a seqüência associada à menor soma dos tempos de fluxo.
- Atualize, com a seqüência S', as L primeiras posições da seqüência S.

Após a apresentação dos métodos heurísticos e a proposição de um novo método heurístico construtivo de duas fases, apresenta-se, na próxima seção, a experimentação computacional.

4. EXPERIMENTAÇÃO COMPUTACIONAL

O método heurístico proposto (S&N) foi comparado com os melhores métodos heurísticos reportados na literatura, ou seja, Raj1 e Raj2, desenvolvidos por Rajendran e Chaudhuri (1990); Bert, de Bertolissi (2000); e o algoritmo heurístico composto de

três fases PH1(p), desenvolvido por Aldowaisan e Allahverdi (2004).

Na experimentação foram avaliados 7.200 problemas, divididos em três grupos. O primeiro grupo de 2.000 problemas, considerados de pequeno porte, subdivididos em 20 classes, de acordo com o número de tarefas n ∈ {5, 6, 7, 8, 9} e o número de máquinas m ∈ {5, 10, 15, 20}.

O segundo grupo envolveu 3.200 problemas, considerados de médio porte, com número de tarefas n ∈ {10, 20, 30, 40, 50, 60, 70, 80} e máquinas m ∈ {5, 10, 15, 20}, totalizando 32 classes de problemas.

O terceiro grupo foi constituído por 2.000 problemas, subdivididos em 20 classes, considerados de grande porte, com número de tarefas n ∈ {90. 100, 110, 120, 130} e m ∈ {5, 10, 15, 20}.

Dessa forma, em cada classe (n,m) foram testados 100 problemas. Os tempos de processamento das operações foram gerados aleatoriamente a partir de uma distribuição uniforme no intervalo [1,99]. Neste trabalho, os métodos foram codificados em linguagem Delphi, e o equipamento utilizado foi um microcomputador Intel Celeron 2,66 GHz, com 256 MB de memória RAM e disco rígido de 40 Gb.

5. MÉTODO DE ANÁLISE

As estatísticas usadas para avaliar o desempenho dos métodos foram a Porcentagem de Sucesso, o Desvio Médio Relativo e o Tempo Médio de Computação. Para facilitar a visualização dos resultados, eles são apresentados em fórmula de gráficos.

A Porcentagem de Sucesso é definida pelo quociente entre o número total de problemas para os quais o método obteve o melhor tempo total de fluxo e o número total de problemas resolvidos. Obviamente, quando os dois métodos obtêm o melhor tempo total de fluxo para o mesmo problema, suas Porcentagens de Sucesso são simultaneamente melhoradas.

O Desvio Relativo (*DRh*) quantifica o desvio que o método *h* obtém em relação ao melhor tempo total de fluxo obtido para um mesmo problema, sendo calculado como se segue:

$$DR_h = \left(\frac{F_h - F_*}{F_*} \right) \qquad (4)$$

em que, F_h = = tempo total de fluxo obtido pelo método *h*; e

F_* = melhor tempo total de fluxo obtido pelos métodos, para determinado problema.

O tempo médio de computação (em segundos) é calculado pela soma dos tempos de processamento em cada problema, dividido pelo número total de problemas resolvidos.

Todos os resultados apresentados foram agrupados com relação ao número de máquinas, possibilitando uma análise global dos resultados.

6. ANÁLISE DOS RESULTADOS

Os resultados foram analisados em três partes, na seguinte ordem: Porcentagem de Sucesso, Porcentagem de Desvio Médio Relativo, e Tempo Médio de Computação. Com o objetivo de facilitar a interpretação dos resultados, são apresentadas as respectivas tabelas e figuras.

Análise da Porcentagem de Sucesso

A Tabela 1 e as Figuras 2, 3 e 4 indicam as Porcentagens de Sucesso em função do porte do problema.

Em uma primeira análise, verificou-se que a porcentagem de sucesso do método S & N é superior à de todos os outros métodos, em todos os casos de variação de 5 a 130 tarefas.

As porcentagens de sucesso da Figura 2 indicam que dois grupos de métodos apresentam desempenhos diferentes para problemas pequenos, o grupo formado pelos métodos PH1(p) e S & N têm comportamento muito próximo com desempenho superior de S & N.

Observando as Figuras 3 e 4, verifica-se uma tendência acentuada de melhor desempenho do método S & N a partir de problemas com 20 tarefas. Após 90 tarefas, o método S & N exibe uma tendência pequena de continuidade de melhoria, já o método PH1(p), uma tendência de estabilização.

Outra observação importante verificada na Tabela 1 são as porcentagens de sucesso dos métodos Raj1, Raj2 e Bert para solução de problemas acima de 100 tarefas, em que se verificou zero porcentagem de sucesso, levando à conclusão de que os métodos são ineficientes para problemas de grande porte.

Por fim, fica consolidado que, independentemente do número de máquinas, o método S & N é superior aos demais, sobretudo para problemas de médio e grande portes.

Tabela 1 - Porcentagem de sucesso para problemas de pequeno, médio e grande portes

Problema	N	Raj1	Raj2	Bert	PH1(p)	S&N
	5	69,75	69,50	71,00	86,50	91,25
	6	62,75	62,75	64,75	87,00	89,50
Pequeno porte	7	51,25	50,00	51,50	75,50	80,25
	8	38,50	34,75	37,75	64,50	70,00
	9	29,50	29,50	30,00	60,25	67,00
	10	24,25	22,00	23,75	53,50	61,00
	20	4,25	7,75	6,00	32,00	52,75
	30	2,50	2,00	1,00	31,00	64,00
Médio porte	40	0,50	1,50	1,25	26,50	70,50
	50	0,25	1,00	0,00	28,25	70,50
	60	0,25	0,50	0,25	25,50	73,50
	70	0,00	0,00	0,25	20,25	79,50
	80	0,00	0,25	0,00	22,50	77,25
	90	0,00	0,25	0,00	19,50	80,25
	100	0,00	0,00	0,00	17,75	82,25
Grande porte	110	0,00	0,00	0,00	14,25	85,75
	120	0,00	0,00	0,00	15,50	84,50
	130	0,00	0,00	0,00	12,75	87,25

Fonte: Elaborada pelos autores.

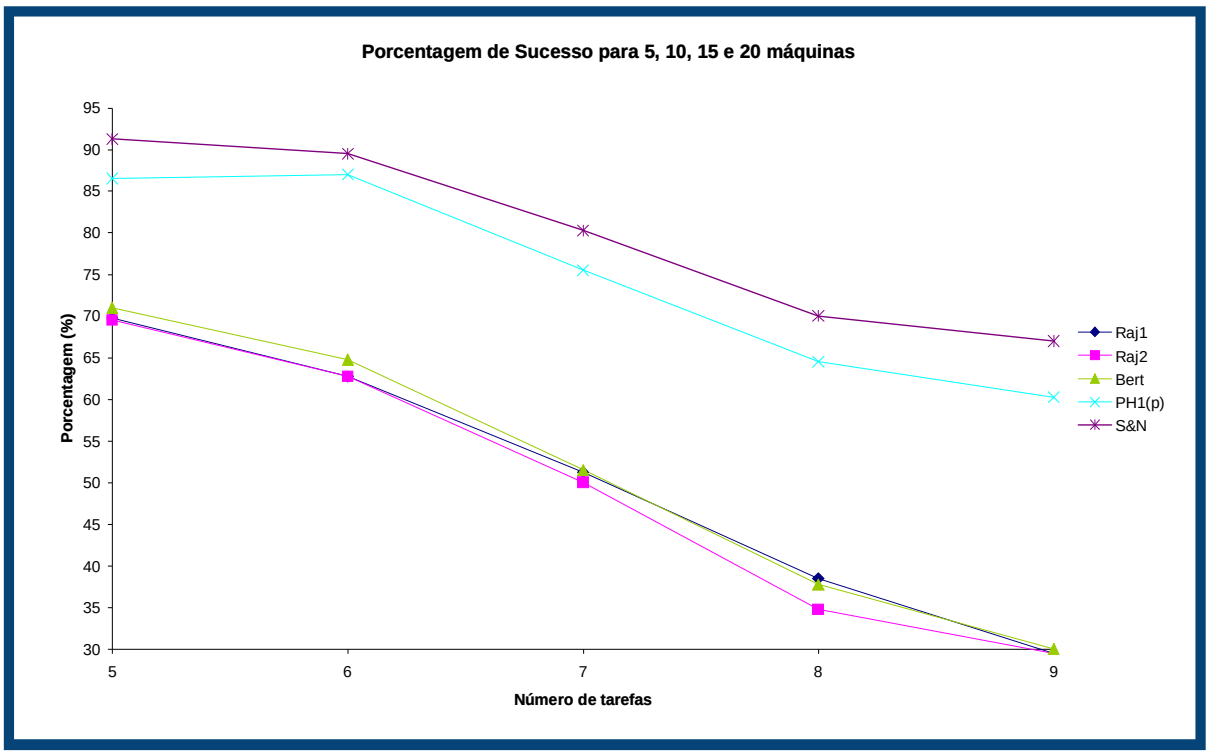

Figura 2 - Porcentagem de sucesso para problemas de pequeno porte.

Fonte: Elaborada pelos autores.

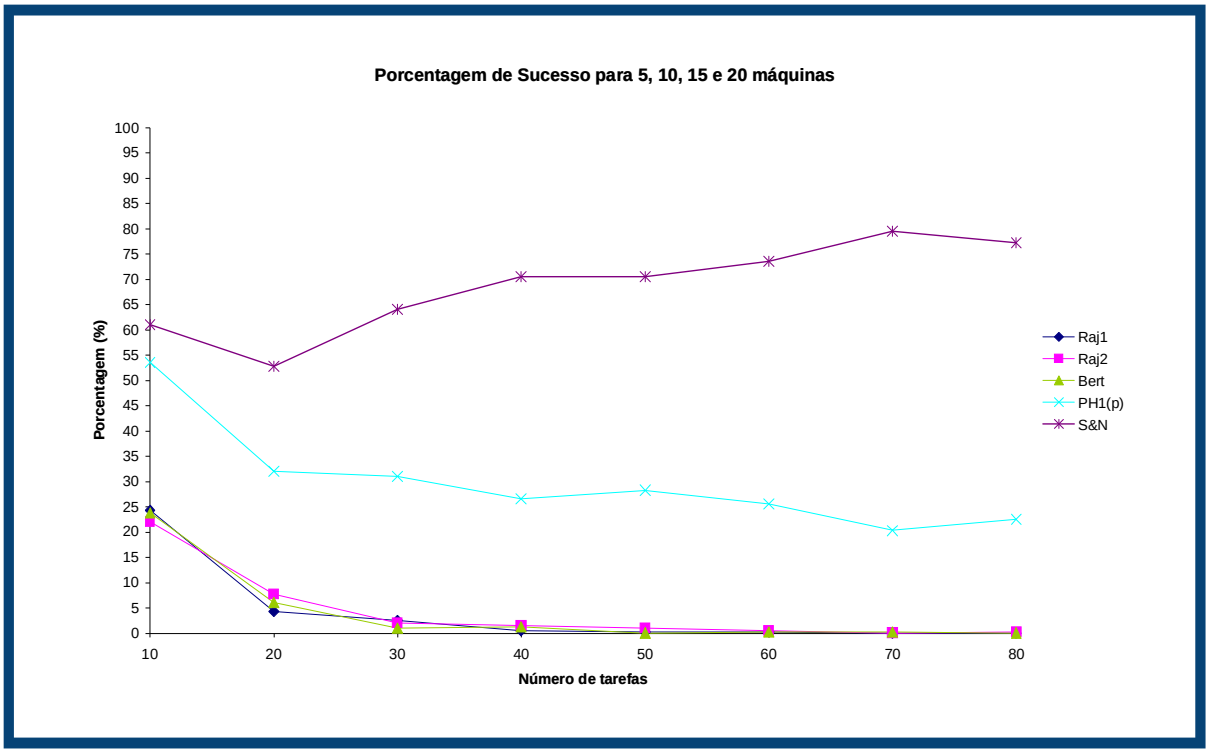

Figura 3 - Porcentagem de sucesso para problemas de médio porte.
Fonte: Elaborada pelos autores.

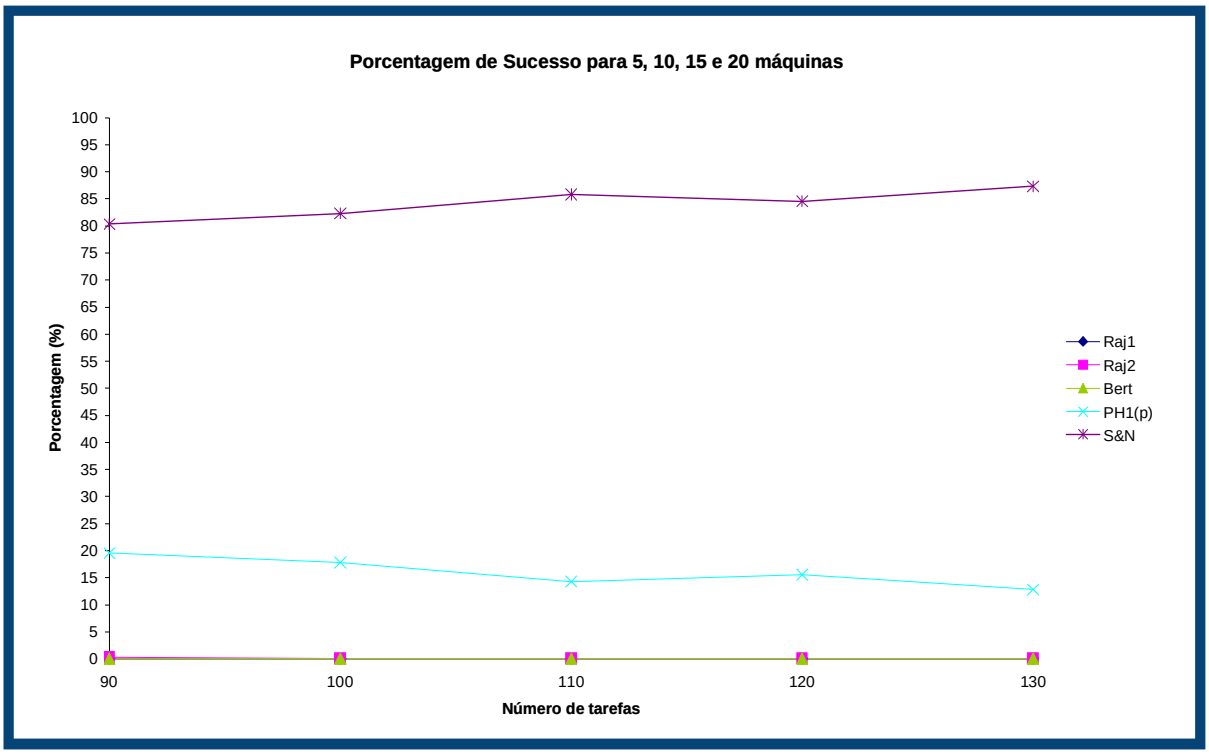

Figura 4 - Porcentagem de sucesso para problemas de grande porte.
Fonte: Elaborada pelos autores.

Análise da Porcentagem de Desvio Médio Relativo

A Tabela 2 e as Figuras 5, 6 e 7 ilustram as Porcentagens de Desvio Médio Relativo em função do porte do problema.

Tabela 2 - Porcentagem de desvio médio relativo para problemas de pequeno, médio e grande portes

Problema	n	Raj1	Raj2	Bert	PH1(p)	S&N
	5	0,27	0,32	0,27	0,09	0,11
	6	0,43	0,55	0,40	0,17	0,10
Pequeno porte	7	0,62	0,63	0,57	0,23	0,23
	8	0,77	0,92	0,77	0,38	0,29
	9	1,02	1,00	0,95	0,39	0,32
	10	1,14	1,21	1,05	0,48	0,44
	20	2,18	2,00	2,05	0,83	0,44
	30	3,01	2,46	2,77	0,80	0,32
Médio porte	40	3,44	2,81	3,35	0,94	0,27
	50	3,77	3,00	3,56	0,94	0,26
	60	3,92	3,05	3,78	0,95	0,18
	70	4,19	3,10	3,88	0,96	0,14
	80	4,37	3,13	4,11	0,99	0,16
	90	4,52	3,20	4,29	1,02	0,12
	100	4,50	3,24	4,20	0,99	0,08
Grande porte	110	4,61	3,13	4,29	1,08	0,06
	120	4,68	3,21	4,41	1,12	0,07
	130	4,82	3,27	4,40	1,13	0,06

Fonte: Elaborada pelos autores.

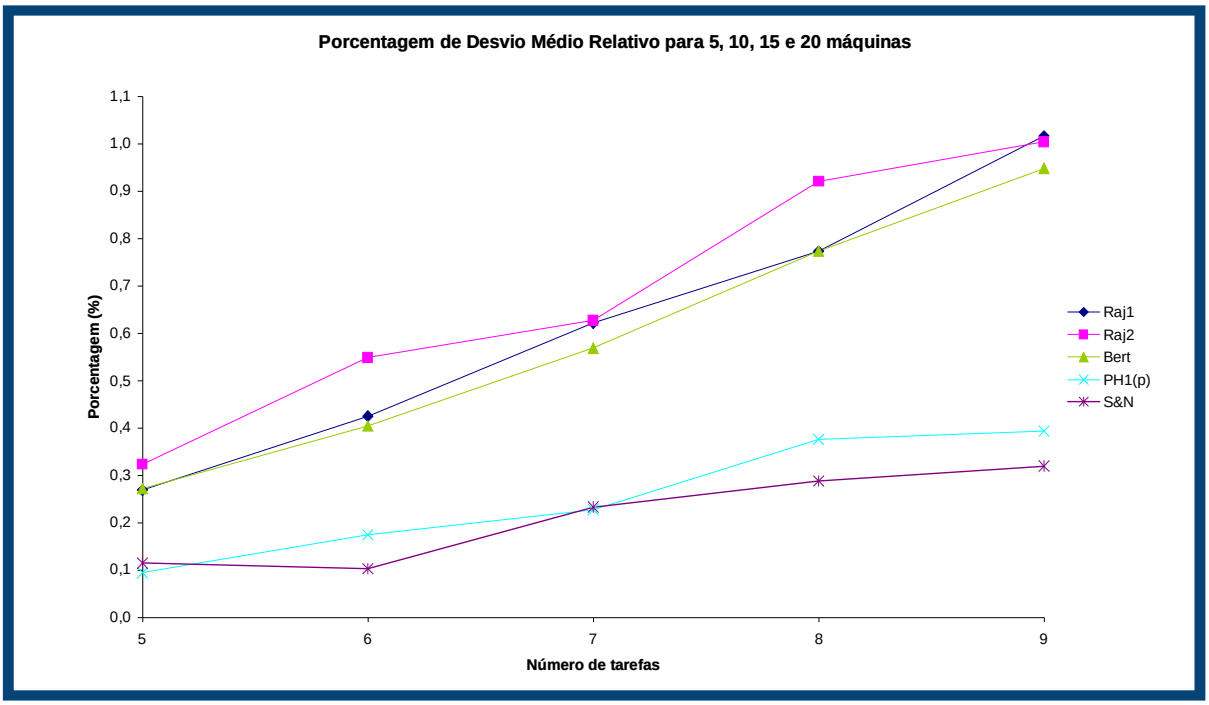

Figura 5 - Porcentagem de desvio médio relativo para problemas de pequeno porte.

Fonte: Elaborada pelos autores.

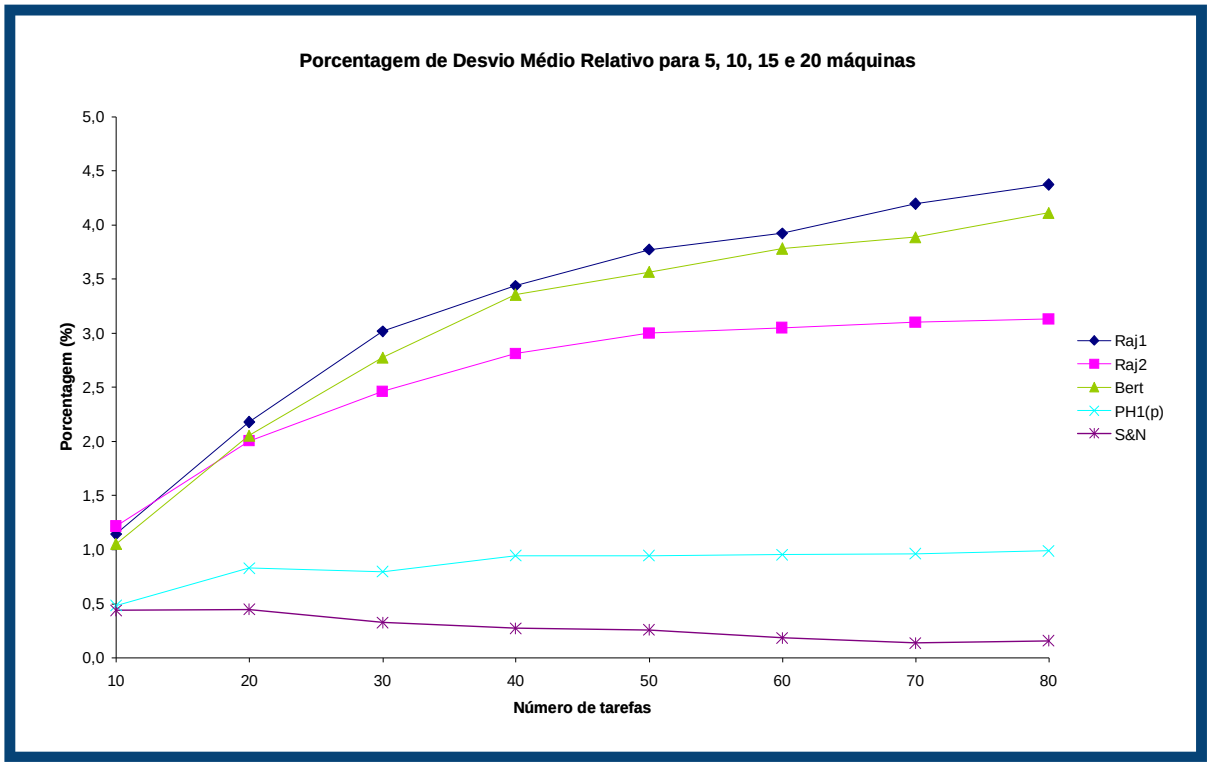

Figura 6 - Porcentagem de desvio médio relativo para problemas de médio porte.

Fonte: Elaborada pelos autores.

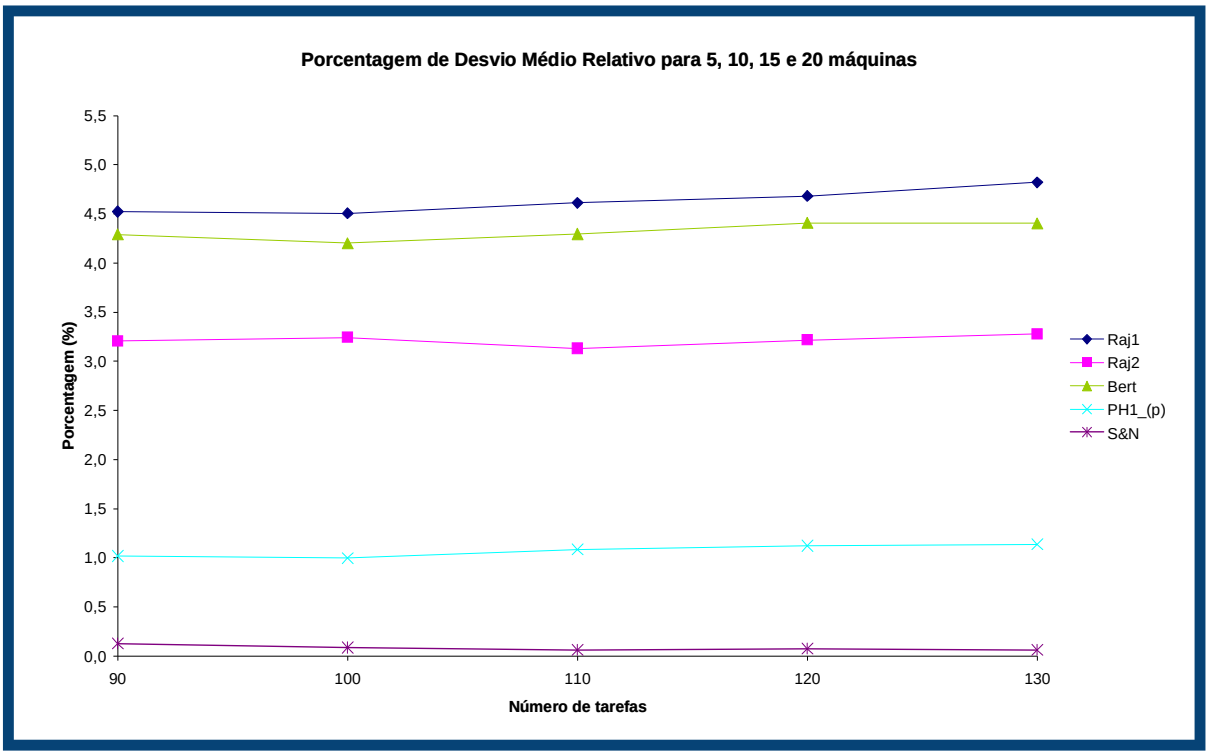

Figura 7 - Porcentagem de desvio médio relativo para problemas de grande porte.

Fonte: Elaborada pelos autores.

O desvio médio relativo para os problemas de pequeno porte apresenta-se muito próximo para os métodos Raj1, Raj2 e Bert, porém os métodos S & N e PH1(p) exibiram menor desvio médio relativo que os demais métodos, principalmente devido ao fato de mostrarem soluções de melhor qualidade.

A partir dos problemas com 20 tarefas, o algoritmo S & N destaca-se, demonstrando um desvio médio relativo inferior ao dos demais algoritmos, principalmente em relação aos algoritmos Raj1, Raj2 e Bert.

Verifica-se, na Figura 7, que, para os problemas de grande porte, a heurística S & N também é a que apresenta menor porcentagem de desvio médio relativo (próximo de zero), enquanto o segundo melhor método exibe uma porcentagem de desvio médio relativo acima de 1%. Também é percebido que, após 90 tarefas, os métodos S & N e PH1(p) apresentam tendência de estabilização dos seus desvios médios, mas ainda com leve tendência de menor desvio do método S & N. Deve-se ressaltar que os valores dos desvios relativos são pequenos, uma vez que o método S & N comprovadamente obtém soluções de alta qualidade.

Tempo Médio de Computação

Na Tabela 3 e nas Figuras 8 e 9, mostram-se os tempos médios de computação em função do porte do problema.

O método S & N para solução dos problemas de grande porte, assim como para os de médio porte, tem característica particular em relação aos demais: quanto maior o número de tarefas, maior o tempo de computação médio. Para problemas com 130 tarefas, o tempo de processamento aproxima-se de 2 s, como pode ser observado na Figura 9.

De forma geral, entre todos os métodos avaliados nesta experimentação, verifica-se que o proposto S & N é o que apresenta melhores resultados, tanto em termos de porcentagem de sucesso quanto em termos de porcentagem de desvio médio relativo.

Percebe-se também que o tempo de computação médio do método proposto S & N possui um crescimento exponencial em relação ao número de tarefas. Porém, mesmo em casos extremos, o método exige um tempo computacional que não inviabiliza sua aplicação.

Tabela 3 - Tempo médio de computação para problemas de pequeno, médio e grande porte

Problema	N	Raj1	Raj2	Bert	PH1(p)	S&N
	5	0,0002	0,0001	0,0005	0,0006	0,0000
	6	0,0002	0,0002	0,0003	0,0002	0,0003
Pequeno porte	7	0,0002	0,0003	0,0006	0,0006	0,0006
	8	0,0004	0,0003	0,0008	0,0010	0,0007
	9	0,0006	0,0007	0,0007	0,0015	0,0014
	10	0,0004	0,0005	0,0010	0,0012	0,0008
	20	0,0006	0,0011	0,0016	0,0031	0,0031
	30	0,0018	0,0020	0,0033	0,0084	0,0107
Médio porte	40	0,0042	0,0040	0,0050	0,0162	0,0219
	50	0,0057	0,0062	0,0080	0,0291	0,0475
	60	0,0090	0,0094	0,0123	0,0475	0,0912
	70	0,0124	0,0139	0,0180	0,0722	0,1627
	80	0,0179	0,0189	0,0238	0,1061	0,2681
	90	0,0251	0,0256	0,0317	0,1453	0,4351
	100	0,0317	0,0331	0,0399	0,1945	0,6432
Grande porte	110	0,0418	0,0430	0,0519	0,2568	0,9485
	120	0,0503	0,0527	0,0616	0,3485	1,4525
	130	0,0657	0,0661	0,0778	0,4392	1,9786

Fonte: Elaborada pelos autores.

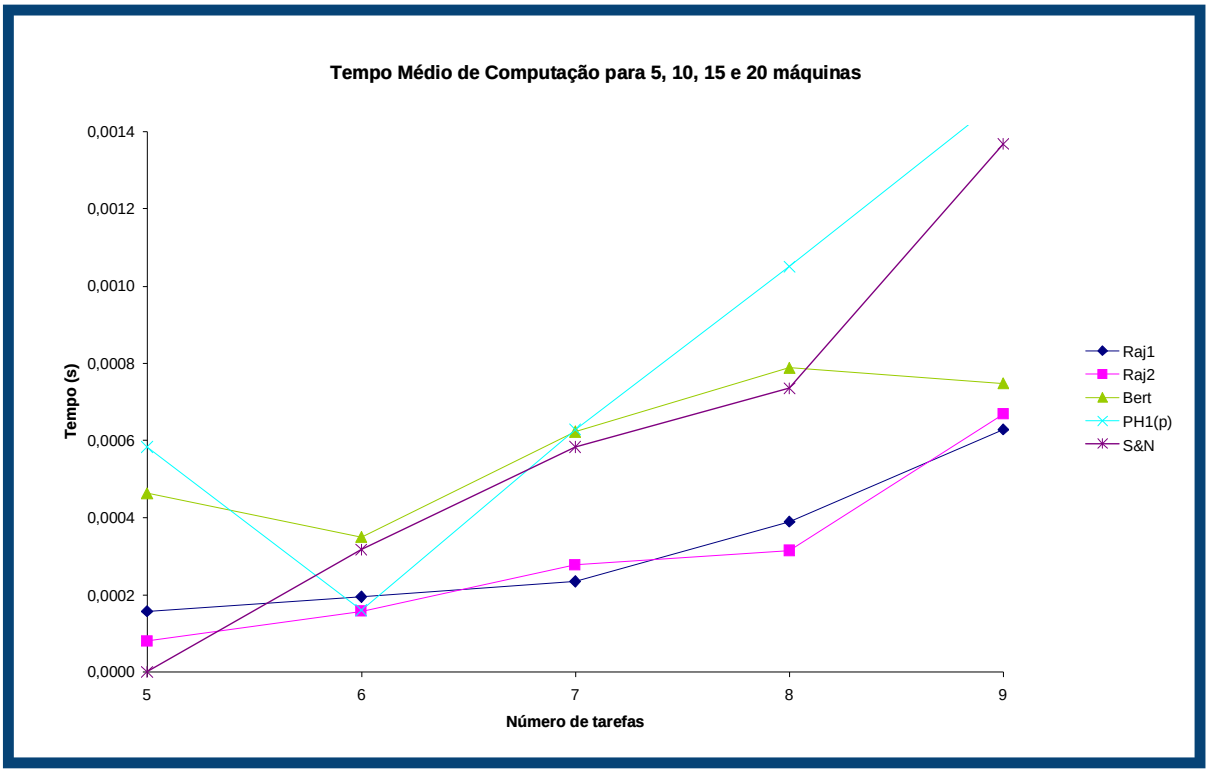

Figura 8 - Tempo médio de computação para problemas de pequeno porte.

Fonte: Elaborada pelos autores.

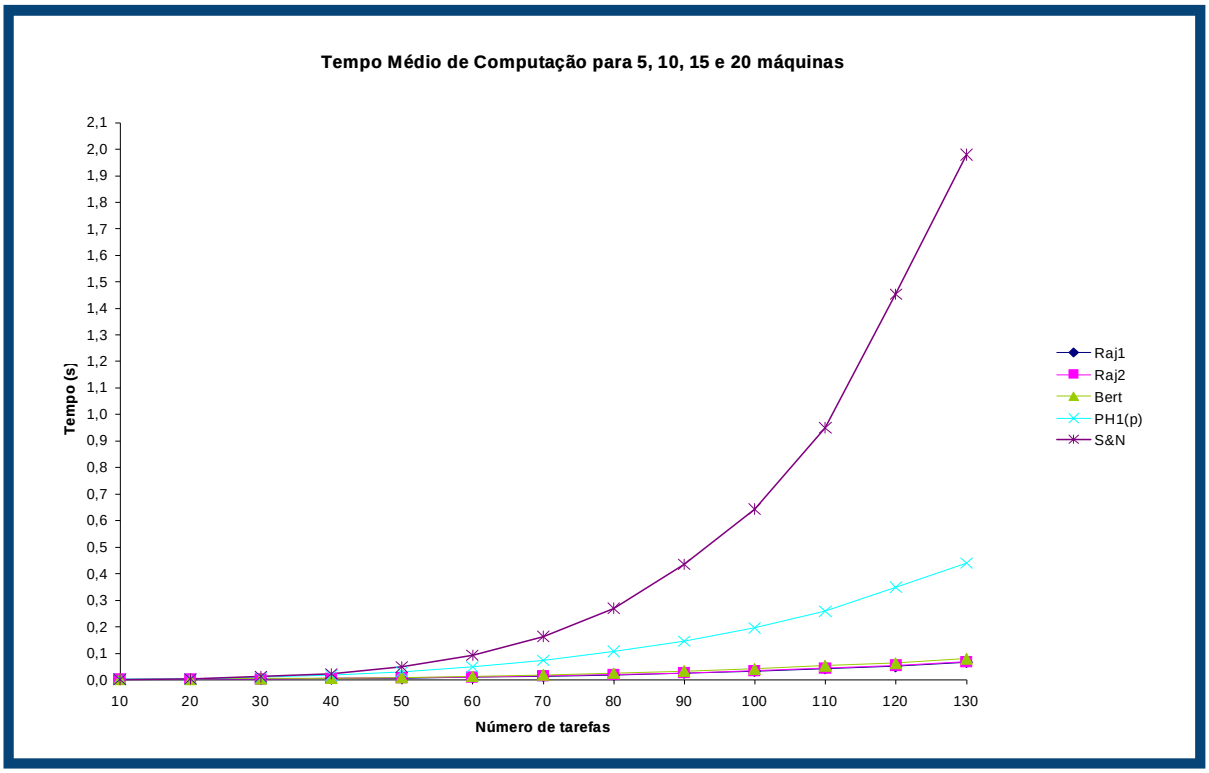

Figura 9 - Tempo médio de computação para problemas de médio e grande porte.

Fonte: Elaborada pelos autores.

7. CONSIDERAÇÕES FINAIS

Preliminarmente, ressalta-se que para fins práticos as soluções obtidas pelos métodos heurísticos de duas e de três fases para o problema *NWFS* são suficientes, ou seja, tal problema pode ser considerado como já resolvido. Porém, tendo em vista a complexidade do problema em questão, a busca por novos métodos que contêm adequado equilíbrio entre a qualidade da solução e a eficiência computacional e a simplicidade e facilidade de implementação ainda continua como uma direção para novas pesquisas. Dessa forma, este trabalho apresentou uma contribuição que procurou evidenciar que, apesar de os algoritmos proporcionarem boas soluções, é possível, por meio da propriedade já existente e apresentada por Bertolissi (2000), criar novos métodos de soluções para o problema.

O principal aspecto a ser destacado neste artigo se refere ao fato de que os resultados experimentais apontaram que o método heurístico proposto S & N possui desempenho superior ao dos demais, para solução do problema em questão.

A experimentação apresentou um tempo computacional maior em relação aos métodos existentes, porém tal aumento é compensado pela facilidade de implementação do método e pela melhoria na qualidade das soluções.

Ainda com relação à experimentação computacional realizada, uma consideração interessante não pode ser esquecida. Quando foram avaliados separadamente os métodos Bert, Raj1 e Raj2, verificou-se que o método proposto por Bertolissi (2000) apenas se equiparava aos propostos por Rajendran e Chaudhuri (1990) para problemas de pequeno porte, sendo inferior nos problemas de médio e grande portes. Assim, diferentemente da conclusão apresentada por Bertolissi (2000), seu método não é superior a Raj1 e Raj2.

8. REFERÊNCIAS

ALDOWAISAN, T.; ALLAHVERDI, A. New heuristics for m-machine no-wait flowshop to minimize total completion time. **OMEGA** - The International Journal of Management Science, v. 32, p. 345-352, 2004.

BERTOLISSI, E. A simple no-wait flowshop scheduling heuristic for the no-wait flow-shop problem. In*:* INTERNATIONAL CONFERENCE ON COMPUTER-AIDED PRODUCTION ENGINEERING, CAPE´99, 15., 1999. **Proceedings**... [S.l.]: University of Durham Publishers, 1999. p. 750-755.

BERTOLISSI, E. Heuristic algorithm for scheduling in the no-wait flow-shop. **Journal of Materials Processing Technology**, v. 107, p. 459-465, 2000.

BONNEY, M. C.; GUNDRY, S. W. Solutions to the constrained flowshop sequencing problem. **Operations Research Quarterly**, v. 24, p. 869-883, 1976.

CHEN, C.; NEPPALLI, R. V.; ALJABER, N. Genetic algorithms applied to the continuous flow shop problem. **Computers Industrial Engineers**, v. 30, n. 4, p. 919-929, 1996.

DEMAN, J. M. V.; BAKER, K. R. Minimizing mean flowtime in the flow shop with no intermediate queues. **AIIE Transactions**, v. 6, n. 1, p. 28-34, 1974.

FINK, A.; VOβ, S. Solving the continuous flow-shop scheduling problem by metaheuristics. **European Journal of Operational Research**, v. 151, p. 400-414, 2003.

FRAMINAN, J. M.; LEISTEN, R. An efficient constructive heuristic for flowtime minimisation in permutation flow shops. **OMEGA** – The International Journal of Management Science, v. 31, p. 311-317, 2003.

KING, R.; SPACHIS, A. S. Heuristics for flowShop scheduling. **International Journal of Production Research**, v. 18, p. 343-357, 1980.

NAWAZ, M.; ENSCORE JR., E. E.; HAM, I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. **OMEGA** – The International Journal of Management Science,v. 11, n. 1, p. 91-95, 1983.

RAJENDRAN, C.; CHAUDHURI, D. Heuristic algorithms for continuous flow-shop problem. **Naval Research Logistics**, v. 37, p. 695-705, 1990.