UM MÉTODO HEURÍSTICO APLICADO NO PROBLEMA DE PROGRAMAÇÃO *FLOW SHOP* PERMUTACIONAL

José Lassance de Castro Silva
Universidade Federal do Ceará-UFC
Campus do Pici, Bloco 910, CEP 60000-000, Fortaleza-CE

Ney Yoshihiro Soma
Instituto Tecnológico de Aeronáutica
Vila das Acácias, CEP 12228-900, São José dos Campos – SP

Resumo

Neste trabalho descrevemos uma nova metodologia aplicada na resolução do problema de programação *flow shop* permutacional, mais conhecido na literatura como *FlowShop Scheduling Problem* (FSP). A metodologia também pode ser aplicada a classe dos Problemas de Otimização Combinatória Permutacional. O método é simples de programar computacionalmente devido a estrutura usada na modelagem do problema. Ele avalia um conjunto fixo de soluções, construído através de permutações, que diversifica a busca dentro do conjunto de soluções viáveis do problema. O FSP pertence à classe dos problemas NP-difícil, que justifica o uso de técnicas refinadas aplicadas na resolução do mesmo com o intuito de encontrar boas soluções viáveis a um baixo custo em termos de recursos computacionais. Extensivos experimentos computacionais foram realizados e reportados para instâncias do problema com até 100 tarefas e 20 máquinas, e os resultados são comparados com aqueles encontrados na literatura.
Palavras-chave: Problema de Seqüenciamento, Otimização Combinatória, Heurística.

Abstract

The aim of this paper is to present a new method applied to solve the FlowShop Scheduling Problem (*FSP*). The method can also be used to solve Combinatorial Optimization Problems that involves permutation, it is a simple program of computer where the solution structure is based on permutation. This technique partitions the set of feasible solutions into small regions to diversify the local search. The technique was applied effectively to the FSP which is an NP-hard problem and difficult to be solved in the practice. Extensive computational experiments are reported for instances with up to 100 jobs and 20 machines and the results are compared with those obtained from the literature.
Key words: Scheduling Problem, Combinatorial Optimization, Heuristics.

1. Introdução

O problema de programação de operações em um ambiente *Flow Shop* é um problema de programação de produção no qual *n* tarefas devem ser processadas por um conjunto de *m* máquinas distintas, tendo o mesmo fluxo de processamento nas máquinas. Usualmente a solução do problema consiste em determinar uma seqüência das tarefas dentre as (*n*!) seqüências possíveis, que é mantida para todas as máquinas (programação permutacional) e que geralmente procura minimizar a duração total da programação (*makespan*), ou seja, o intervalo de tempo entre o início de execução da primeira tarefa na primeira máquina e o término de execução da última tarefa na última máquina.

O Problema de Programação *Flow Shop* Permutacional, denominado na literatura de *FlowShop Scheduling Problem* (FSP), possuí as seguintes característica:

a) É dado um conjunto com *n* tarefas $J_1, J_2, ..., J_n$;
b) É dado um conjunto com *m* máquinas $M_1, M_2, ..., M_m$;
c) Cada tarefa demanda *m* operações, com uma operação representando o tempo de processamento da tarefa por máquina;
d) As tarefas seguem o mesmo fluxo de operações nas máquinas, isto é, para qualquer $j = 1,2,..., n$, a tarefa J_j deve ser processada primeiro na máquina M_1 , depois na máquina M_2, e assim por diante até a última máquina, no caso máquina M_m;
e) Caso a tarefa Jj não utilize todas as máquinas, o seu fluxo continua sendo o mesmo, todavia com o tempo de ocupação sendo igual a zero;
f) Uma máquina pode processar somente uma operação de cada vez, e iniciada uma operação, ela deva ser processada até a sua conclusão;
g) O número de seqüências distintas possíveis para realização das tarefas nas máquinas é grande, i. e. $O(n!)$.

Um *input* FSP é dado por *n*, *m* e uma matriz $P(n \text{ x } m)$ de elementos não negativos, onde P_{ij} denota o tempo de processamento da tarefa J_j na máquina M_i. Seguindo os 4 parâmetros da notação *A*/*B*/*C*/*D* adotada por Conway *et al.* (1967), o problema é classificado como $n/m/P/F_{max}$. Na recente notação paramétrica $\alpha/\beta/\gamma$, proposta por Graham *et al.* (1979), o problema é denotado como sendo $F/\text{prmu}/C_{max}$. O FSP pertence à classe dos problemas NP-*hard*, quando $m \geq 3$, conforme Garey *et al*. (1976), de forma que pode ser resolvido eficientemente de maneira ótima somente em casos de pequeno porte. No caso em que $m=2$, o problema pode ser solucionado através de um algoritmo em tempo polinomial, Johnson (1954).

Gupta e Stafford Jr. (2006), fizeram um levantamento científico sobre o problema nas últimas cinco décadas e, constataram que existem mais de 1200 artigos, na literatura sobre pesquisa operacional, contendo vários aspectos deste problema. Os métodos de resolução exata, geralmente, são aplicados a problemas de pequena instância ($n \leq 20$), e mesmo neste caso o tempo computacional ainda é muito alto. Ruiz *et al.* (2006) aborda, de forma resumida, alguns métodos utilizados na resolução do problema de forma aproximativa, através de heurísticas e meta-heurísticas. Dentre elas, podemos citar as heurísticas: PAL, Palmer (1965); CDS, Campbell *et al.* (1970); que serviram como base para a comparação de desempenho da metodologia adotada. Técnicas de Programação Matemática, tais como Programação Linear Inteira, Selen e Hott (1986), Wilson (1989), e técnicas de enumeração do tipo *branch-and-bound*, Ignal e Schrage (1965), Potts (1980), têm sido empregadas para a solução ótima do problema. Entretanto, tais técnicas não são eficientes em termos computacionais, em problemas de médio e grande porte. Desta forma, justifica-se a utilização de métodos heurísticos usados na resolução do problema.

Os métodos heurísticos podem ser classificados como construtivos ou melhorativos, conforme descrito em Moccellin (1999). No caso dos métodos construtivos, a seqüência adotada como solução do problema é obtida:

- Diretamente a partir da ordenação das tarefas segundo índices de prioridade calculados em função dos tempos de processamento das tarefas, como por exemplo: Palmer (1965) e Gupta (1971);

- ou escolhendo-se a melhor seqüência das tarefas a partir de um conjunto de seqüências também obtidas utilizando-se índices de prioridade associados às tarefas. Neste caso, podem ser citados: Campbell *et al.* (1970) e Hundal e Rajgopal (1988);
- ou ainda, a partir da geração sucessiva de seqüências parciais das tarefas (subseqüências) até a obtenção de uma seqüência completa através de algum critério de inserção de tarefas, como por exemplo o método NEH, descrito em Nawaz *et al.*, (1983).

No caso dos métodos melhorativos, obtém-se uma solução inicial e posteriormente através de algum procedimento iterativo (geralmente envolvendo trocas de posições das tarefas na seqüência) busca-se obter uma seqüência das tarefas melhor que a atual quanto à medida de desempenho adotada.

A dificuldade de solucionar o FSP de forma exata está no grande número de soluções existentes para as instâncias de médio e grande porte. Problemas desta natureza são comumente abordados através de Métodos Heurísticos Construtivos (heurísticas), que são procedimentos que permitem determinar uma ou mais soluções para determinados problemas que muitas vezes se apóiam em uma abordagem intuitiva, na qual a estrutura particular do problema seja considerada e explorada de forma inteligente, para a obtenção de uma boa solução adequada. Conforme Reeves (1995), as heurísticas foram desenvolvidas com a finalidade de resolver problemas de elevado nível de complexidade em tempo computacional razoável. Ao se pensar em um problema combinatório complexo, uma opção seria analisar todas as combinações possíveis para conhecer a melhor delas. Whitley *et al.* (1991) fizeram uma boa abordagem da relação do problema de seqüenciamento com o problema do caixeiro viajante, onde constataram que as técnicas utilizadas na resolução de um dos problemas também podem ser aplicadas na resolução do outro, com pequenas modificações.

O ataque ao FSP sugerido aqui se baseia numa abordagem dos métodos heurísticos construtivos. O nosso principal objetivo é mostrar uma nova abordagem de resolução para o FSP, denominada de Heurística Permutacional de Parâmetro *k* (*HPk*), que sem muito sacrifício também pode ser utilizado na resolução dos problemas de otimização combinatória que envolve permutação. Os experimentos realizados com *HPk* são descritos adiante, onde pode-se concluir o seu desempenho. Pretende-se verificar qual a influência na qualidade dos resultados obtidos comparados com outros métodos heurísticos construtivos, encontrados na literatura, que usaram as mesmas classes de problemas experimentais.

A organização do artigo segue à: Na Seção 2, apresentaremos as principais idéias de HPk e descrevemos o FSP como um Problema de Otimização Combinatória Permutacional. Na Seção 3, serão apresentados os experimentos computacionais, com os resultados obtidos. Finalizamos nosso trabalho com a apresentação dos melhoramentos ora sendo desenvolvidos, Seção 4, enquanto na Seção 5 tem-se o material bibliográfico consultado.

2. O Método Heurístico

Um Problema de Otimização Combinatória Permutacional (*POCP*) pode ser definido por um terno (S, g, n), onde S é o conjunto de todas as soluções viáveis (soluções que satisfazem as restrições do problema, com $|S| = n!$), g é a função objetiva que aplica a cada solução $s \in S$ um número real e n é uma instância do problema. O objetivo é encontrar a solução $s \in$ S que minimize a função objetiva g. Podemos representar s como uma permutação de n elementos distintos, ou seja, $s = < a_1\ a_2 \ldots a_n >$. N($s$) é chamada *a vizinhança de s* e contém todas as soluções que podem ser alcançadas de s por um simples movimento. Aqui, o significado de um *movimento* é aquele de um operador que transforma uma solução para uma outra com pequenas modificações.

Silva e Soma (2001) desenvolveram uma técnica para resolver POCP, denominada Heurística Permutacional (*HP*). Através desta técnica conseguimos intensificar o processo de diversificação, aumentando o número de soluções a ser avaliada no problema. O procedimento *HP* consiste basicamente em dividir o conjunto de soluções viáveis S em n vizinhanças N(s_i) distintas entre si, onde para cada $1 \leq i \leq n$ s_i é uma permutação que inicia com o elemento i e, cada uma destas vizinhanças será particionada em quatro subvizinhanças $N(s_{ij}) \subset N(s_i)$, $1 \leq j \leq 4$, com s_{ij} definido adiante. As soluções (permutações) s pertencentes a $N(s_i)$ também iniciam com o elemento i. Desta forma $N(s_i) \cap N(s_k) = \emptyset$, $1 \leq i,k \leq n$, com $i \neq k$. Exceto $s_{i1} = s_i$, as quatro permutações s_{i1}, s_{i2}, s_{i3} e s_{i4} que produzirão as quatro subvizinhanças $N(s_{i1})$, $N(s_{i2})$, $N(s_{i3})$ e $N(s_{i4})$, respectivamente, são obtidas da troca de posições dos elementos de s_i, da seguinte forma:

1°) A permutação s_{i2} mantém a primeira posição de s_i e inverte as (n-1) posições restantes de s_i;
2°) A permutação s_{i3} mantém a primeira posição de s_i e troca seqüencialmente 2 a 2 as demais posições adjacentes de s_i;
3°) A permutação s_{i4} mantém a primeira posição de s_{i3} e inverte as (n-1) posições restantes de s_{i3}.

A subvizinhança $N(s_{ij})$ é formada por todas as permutações que são obtidas de s_{ij} trocando de posição 2 a 2 todos os elementos de s_{ij}, a partir da segunda posição, e mantendo a ordem dos outros elementos inalterado. Assim, o número de permutações em cada subvizinhança é $[\ (n-1)\times(n-2)/2 + 1\ ]$ que implica em $[\ 4\times[(n-1)\times(n-2)/2 +1]\ ]$ permutações em cada vizinhança e um total de $[\ 2 \times n \times (n-1) \times (n-2)\ +\ 4n\ ]$ permutações geradas para o problema. Dada uma permutação qualquer s_1, podemos obter os demais s_i, $2 \leq i \leq n$, trocando somente o elemento da primeira posição de s_1 pelo elemento da i-ésima posição. Sem perdas de generalidades podemos supor s_1=< 1 2 3 ... n >, então s_2=< 2 1 3 4 5 ... n >, s_3=< 3 2 1 4 5 ... n >, ..., s_n=< n 2 3 4 5 ... (n-1) 1 >.

HPk Adota o seguinte critério, para avaliar um conjunto de soluções construídas a partir de algumas soluções configuradas em HP. Seja s uma das permutações do tipo s_{i1}, s_{i2}, s_{i3} e s_{i4}, com i entre 1 e n, definida anteriormente. Sem perda de generalidade, a permutação s pode ser vista com a seguinte forma s=< i a_2 ... a_n >. A partir da segunda posição de s dividimos os seus (n-1) elementos em k grupos G_1, G_2, ..., G_k, cada um deles tendo: os elementos de s na mesma ordem em que os mesmos se encontram em s; e no máximo $\lceil (n-1)/k \rceil$ elementos. O último grupo, G_k, terá exatamente r elementos, onde r é o resto da divisão de n-1 por k, ou seja, $r=(n-1)$ módulo k. Em seguida, geram-se k! novas soluções s', fazendo:

1º) coloque i na primeira posição de s';
2º) complete o restante das posições de s' com uma das k! permutações distintas dos grupos G_1, ..., G_k.

Este procedimento é executado para cada uma das $4 \times n$ permutações distintas de HP, que geram as subvizinhanças. Desta forma, o número total de permutações a serem avaliadas em HPk será de $4\times n\times(k!)$, pois para cada permutação gerada em *HP*, que gera uma subvizinhança, foram geradas (k!) permutações, inclusive a própria permutação que gerou as novas soluções.

Exemplificando o procedimento, tem-se que, se s= < 4 5 3 1 7 2 8 6 > e k=3, então os três grupos seriam G_1=< 5 3 1 >, G_2= < 7 2 8 > e G_3=< 6 >, e as novas soluções estariam assim configuradas < 4 <u>5 3 1</u> <u>7 2 8</u> <u>6</u> >, < 4 <u>5 3 1</u> <u>6</u> <u>7 2 8</u> >, < 4 <u>7 2 8</u> <u>5 3 1</u> <u>6</u> >, < 4 <u>7 2 8</u> <u>6</u> <u>5 3 1</u> >, < 4 <u>6</u> <u>5 3 1</u> <u>7 2 8</u> > e < 4 <u>6</u> <u>7 2 8</u> <u>5 3 1</u> >.

O valor de k está definido entre 1 e (n-1). Para k=1, HPk irá gerar somente as permutações de HP que geram as subvizinhanças e, neste caso, HP avaliará muito mais soluções que HPk. No caso em que $k=(n-1)$, HPk não só avaliará todas as soluções possíveis e distintas para o problema como também repetirá três vezes mais o número total de soluções distintas existentes. Pois, o número total de soluções avaliadas seria de $4\times n\times((n-1)!)=4\times(n!)$, enquanto existem n! soluções distintas possíveis. Para HPk avaliar todas as n! soluções distintas possíveis do problema, o valor de k teria que satisfazer a seguinte condição: $4\times n\times(k!)=(n!)$, isto é, $k! = (n-1)! / 4$. O procedimento para gerar as soluções de HPk, é dado a seguir:

```
for(i=1; i<=n; i++) s1[i]=i;      /* s1 é a 1ª solução que gera cada 1 das 4 soluções sij */
```

```
for(i=1; i<=n; i++) {              /* p1 identifica a primeira posição de s */
   if (i>1) { s1[1]=i; s1[i]=1; }
   for(j=1; j<=4; j++) {           /* Procedimento para gerar as 4 soluções sij  */
      switch (j) {
             case 1: for(k=1; k<=n; k++) s2[k]=s1[k]; break;
             case 2: p=(n+1)/2; for(k=2; k<=p; k++) {
                       s=s2[k]; s2[k]=s2[n+2-k]; s2[n+2-k]=s; } break;
             case 3: for(k=1; k<=n; k++) s2[k]=s1[k]; k=2;
                       while (k<=(n-1)) { s=s2[k]; s2[k]=s2[k+1]; s2[k+1]=s; k=k+2; }
                       break;
             case 4: p=(n+1)/2; for(k=2; k<=p; k++)
                       { s=s2[k]; s2[k]=s2[n+2-k]; s2[n+2-k]=s; } break;
      }
      Permuta( k, s2);          /* k é o número de grupos e s2 é a solução que gera as k! soluções  */
   }
   if (i>1) { s1[1]=1; s1[i]=i; }
}
```

Permuta é o procedimento usado para determinar as novas soluções a partir de uma solução *s* dada, permutando todos os *k* grupos.

O FSP pode ser modelado como um POCP *P* = (*S*, *g*, *n*), tendo a seguinte forma:

a) Um elemento $s=< J_1\ J_2\ ...\ J_n >$ do conjunto de soluções viáveis *S* é representado por uma permutação das *n* tarefas, com a ordem de *s* determinando a seqüência na qual as tarefas serão processadas;

b) O procedimento *g*, dado a seguir, determina o valor do tempo gasto (*tg*) para processar a seqüência *s*, mais precisamente tem-se que *tg* é o tempo utilizado no processamento da última tarefa de *s* na última máquina M_m.

```
Entrada: m, n, permutação s, Matrizes T(m x n) e P(m x n).
Saída: tg  (tempo gasto para processar todas as n tarefas usando a seqüência s)

for(i=1; i<=m; i++)
          for(j=1; j<=n; j++)  t[i][j]=0;
for(j=1; j<=n; j++) {
          for(i=1; i<=m; i++) {
                    if (i==1) {
                              if (j>=2) t[1][s[j]]=t[1][s[j-1]]+p[1][s[j-1]];
                    } else {
                              if (j==1)
                                        {t[i][s[1]]=t[i-1][s[1]]+p[i-1][s[1]];
                              } else {
                                        x=t[i][s[j-1]]+p[i][s[j-1]];
                                        y=t[i-1][s[j-1]]+p[i-1][s[j-1]];
                                        if (x>=y) t[i][s[j]]=x; else t[i][s[j]]=y;
                              }
                    }
          }
}
tg=t[m][s[n]] + p[m][s[n]];
```

O elemento t_{ij}, da matriz *T* com ordem $m \times n$, representa o tempo para iniciar a tarefa J_j na máquina M_i.

3. Experimentos Computacionais

As heurísticas *HP* e *HPk* foram executadas num micro-computador PC-Asus (RAM de 256Mb e 1.8 GHz) e o código foi implementado em linguagem ANSI C. A Tabela 1, a seguir, mostra o desempenho do método com instâncias da OR-Library (Beasley, 1990), conjunto de dados de Taillard (1993). Nela encontram-se: as classes avaliadas baseadas nos valores de *m* e *n*; os desvios médios de PAL, CDS, HP e HPk, para *k*=3, 4, 5, 6, 7, 8; e o valor ótimo z*. O desvio é dado por $100 \times (z - z^*)/z^*$, onde *z* é o valor de *g* na melhor solução encontrada e *z** é o valor ótimo. Nas últimas linhas da tabela, também são apresentados os valores mínimos e máximos para os desvios, obtidos nos experimentos.

m, n	*z**	PAL	CDS	HP	k=3	k=4	k=5	k=6	k=7	k=8
5, 20	1278	8,29	8,76	4,93	9,08	7,20	3,60	3,60	4,15	3,60
5, 20	1359	5,89	4,78	3,24	5,45	5,45	2,21	2,13	2,13	1,91
5, 20	1081	7,49	15,54	11,38	12,77	10,73	12,12	11,75	10,64	10,64
5, 20	1293	15,24	9,67	11,83	14,54	12,14	11,91	10,75	11,06	10,21
5, 20	1235	10,12	7,13	9,47	10,20	8,83	7,69	7,69	7,85	5,83
5, 20	1195	12,47	9,79	13,47	16,82	11,63	9,21	7,45	4,69	4,69
5, 20	1239	12,99	12,43	7,18	7,75	6,78	7,43	4,04	2,99	2,99
5, 20	1206	8,87	11,19	10,12	12,60	12,27	11,19	8,87	9,20	8,04
5, 20	1230	15,93	10,57	8,13	14,96	12,11	12,85	10,16	10,08	8,21
5, 20	1108	10,92	5,05	10,92	16,06	14,62	11,01	10,47	10,56	6,95
10, 20	1582	13,15	11,06	12,83	19,03	13,84	14,03	10,94	8,79	8,79
10, 20	1659	17,42	11,75	8,62	12,18	12,18	12,18	10,49	10,49	10,49
10, 20	1496	15,57	9,96	12,83	12,83	15,71	11,56	11,63	9,49	9,16
10, 20	1377	15,11	12,35	13,44	18,23	17,57	12,93	13,58	10,02	8,50
10, 20	1419	16,14	9,80	15,57	19,94	14,52	15,36	15,64	12,33	11,77
10, 20	1397	9,31	13,89	13,89	19,83	15,39	9,66	12,88	13,03	8,30
10, 20	1484	16,91	9,84	14,69	10,51	11,86	11,19	7,08	8,02	6,20
10, 20	1538	14,63	16,25	16,84	17,62	16,58	12,87	11,83	10,27	10,27
10, 20	1593	15,25	7,97	9,92	12,99	11,80	9,98	8,47	8,16	7,03
10, 20	1591	19,30	18,42	13,89	10,69	11,94	12,07	10,37	9,62	7,48
20, 20	2297	22,68	11,41	7,71	9,27	10,01	7,79	6,88	6,49	5,75
20, 20	2099	11,05	8,86	11,77	14,39	12,91	11,86	11,24	7,86	6,72
20, 20	2326	15,13	10,28	7,39	10,66	10,32	7,57	8,43	5,76	5,33
20, 20	2223	18,26	9,49	11,88	12,24	10,48	10,84	10,84	7,69	6,61
20, 20	2291	18,03	9,38	11,79	13,09	10,21	9,25	8,95	6,59	6,42
20, 20	2226	15,54	8,81	10,20	13,66	11,64	10,29	10,20	10,69	9,12
20, 20	2273	8,05	9,50	11,88	11,44	10,47	9,68	7,35	7,00	7,00
20, 20	2200	10,68	7,36	11,50	14,23	10,82	11,82	10,23	9,64	8,32
20, 20	2237	23,11	7,91	11,31	15,74	13,68	12,47	11,04	10,37	9,70
20, 20	2178	20,89	13,36	15,52	17,31	11,39	13,50	12,08	13,13	9,87
5, 50	2724	1,84	3,38	4,88	7,97	6,28	4,92	5,29	2,57	3,16
5, 50	2834	7,30	6,99	4,94	7,62	7,73	6,39	6,14	5,47	4,98
5, 50	2621	5,95	3,13	5,61	7,44	8,70	5,72	4,43	3,59	5,04
5, 50	2751	3,96	4,83	6,94	9,74	7,74	5,85	6,69	4,83	4,73
5, 50	2863	3,49	6,11	5,38	7,27	6,74	6,01	5,69	5,83	3,77
5, 50	2829	9,23	7,14	6,61	7,60	9,30	4,88	5,97	5,87	4,77
5, 50	2725	4,40	8,95	6,97	7,16	8,44	7,63	7,01	6,57	5,14
5, 50	2683	5,33	5,67	10,59	10,47	10,10	7,38	7,79	6,26	6,19
5, 50	2552	7,09	9,09	6,11	9,25	7,99	7,21	7,25	6,43	5,68
5, 50	2782	4,78	5,75	7,44	10,14	8,63	6,72	7,66	3,81	4,53
10, 50	2991	16,28	14,38	17,42	15,45	16,55	16,45	13,01	14,68	12,37

10, 50	2867	15,56	13,22	15,70	18,84	18,84	18,84	17,23	16,01	15,17
10, 50	2839	16,98	15,53	19,51	21,98	17,93	19,37	17,29	16,66	16,70
10, 50	3063	14,63	10,77	12,41	14,20	14,46	14,46	11,85	13,16	11,72
10, 50	2976	15,15	13,41	18,48	17,67	17,74	15,52	14,52	12,60	13,31
10, 50	3006	10,55	13,11	16,63	20,09	17,00	15,77	16,87	12,94	14,87
10, 50	3093	11,77	13,81	15,55	15,29	18,01	13,13	13,26	13,22	13,90
10, 50	3037	10,50	11,52	15,90	19,39	17,12	16,00	15,38	12,61	12,25
10, 50	2897	17,85	12,22	18,23	20,99	21,02	17,88	17,47	14,33	14,67
10, 50	3065	11,06	11,88	18,66	18,92	17,23	14,78	14,68	14,26	12,82
20, 50	3771	13,29	14,77	21,06	21,88	20,98	21,69	19,46	18,35	17,69
20, 50	3668	17,31	14,94	18,87	21,51	18,48	18,46	17,45	15,95	16,33
20, 50	3591	17,24	16,65	20,11	24,76	20,05	20,52	19,63	17,40	18,10
20, 50	3635	16,45	17,74	25,25	26,11	22,42	19,17	21,27	19,12	19,28
20, 50	3553	23,16	16,01	19,34	22,94	22,94	21,93	19,25	19,45	19,34
20, 50	3667	17,59	16,36	18,30	20,78	19,80	19,77	17,89	18,03	17,15
20, 50	3672	17,27	12,58	22,17	24,70	22,19	23,07	20,15	19,44	18,65
20, 50	3627	18,83	17,51	23,24	21,89	22,22	20,79	16,96	19,35	19,08
20, 50	3645	24,75	15,56	22,41	23,57	23,37	21,56	20,14	19,31	18,93
20, 50	3696	13,56	15,53	24,95	20,70	21,97	22,13	20,10	20,32	19,21
5, 100	5493	4,66	1,80	3,60	5,28	3,80	4,10	4,19	2,55	2,38
5, 100	5268	0,91	5,60	6,57	6,85	6,25	2,90	3,21	3,19	2,33
5, 100	5175	2,90	6,14	7,75	7,85	6,51	5,89	5,26	4,91	4,02
5, 100	5014	0,70	5,17	7,08	4,67	6,86	5,13	3,19	3,93	4,21
5, 100	5250	1,28	4,02	9,64	8,53	6,59	6,29	4,80	4,30	4,90
5, 100	5135	2,71	2,41	5,63	5,39	6,23	6,39	4,97	3,91	3,72
5, 100	5246	2,48	5,93	7,89	9,23	6,84	5,64	5,22	4,02	3,35
5, 100	5034	4,55	7,01	8,82	8,22	8,52	5,72	5,98	3,73	4,97
5, 100	5448	2,90	5,69	7,95	9,18	7,67	4,53	5,75	4,33	4,31
5, 100	5322	1,97	7,53	6,86	8,61	7,27	7,44	5,94	5,51	4,13
10, 100	5770	6,78	7,61	14,49	12,91	12,77	11,53	10,95	10,87	9,67
10, 100	5349	10,10	9,80	16,06	17,07	16,15	13,80	15,65	13,54	13,97
10, 100	5676	7,95	6,13	12,40	11,56	12,23	12,35	10,78	10,84	9,80
10, 100	5781	9,20	10,31	16,62	17,23	16,33	13,82	12,77	12,23	10,81
10, 100	5467	11,03	10,08	14,87	15,84	14,82	15,20	13,74	12,58	12,90
10, 100	5303	10,69	8,32	15,39	15,01	14,80	13,22	12,11	11,26	11,71
10, 100	5595	15,14	10,83	12,89	14,51	11,97	11,39	11,74	8,79	9,04
10, 100	5617	9,81	10,98	12,89	13,39	12,93	13,00	11,77	11,64	9,63
10, 100	5871	3,58	8,14	11,77	13,40	11,24	11,28	10,85	9,96	8,38
10, 100	5845	7,08	9,27	11,91	11,04	9,27	10,61	9,24	9,84	8,28
20, 100	6106	15,87	13,33	22,57	24,03	21,98	22,40	22,22	20,36	19,21
20, 100	6183	14,15	12,84	18,76	20,39	20,46	19,55	19,04	17,11	16,08
20, 100	6252	15,50	15,63	20,73	19,56	20,76	18,09	19,00	17,26	16,79
20, 100	6254	12,55	12,92	19,28	19,09	18,36	17,51	17,48	15,06	14,41
20, 100	6262	15,92	13,59	19,45	21,70	19,40	17,97	16,80	17,37	15,78
20, 100	6302	12,81	15,57	19,22	20,72	20,37	19,42	19,74	18,03	18,15
20, 100	6184	17,71	15,57	24,18	23,98	21,73	20,50	19,26	19,24	18,68
20, 100	6315	19,73	14,57	25,10	20,22	21,22	19,07	18,64	18,65	16,85
20, 100	6204	17,20	15,99	23,16	23,31	23,74	21,70	21,07	19,94	18,52
20, 100	6404	14,07	11,87	20,61	20,30	18,97	18,22	17,50	16,38	16,40
Média		11,88	10,51	13,42	14,79	13,70	12,50	11,71	10,76	10,10
Mínimo		0,70	1,80	3,24	4,67	3,80	2,21	2,13	2,13	1,91

Máximo	24,75	18,42	25,25	26,11	23,74	23,07	22,22	20,36	19,34

Tabela 1 – Resultados obtidos.

As Figuras 1 e 2 fornecem uma amostra mais detalhada dos resultados, mostrando mais precisamente o comportamento de cada métodos, através dos valores dos desvios obtidos com os experimentos computacionais.

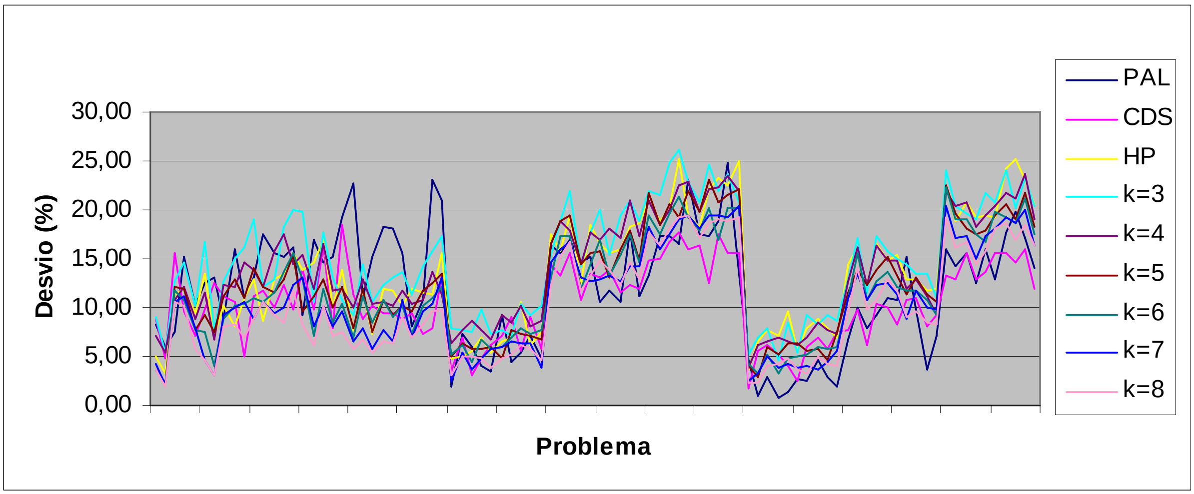

Figura 1 – Representação gráfica das soluções, com base nos desvios.

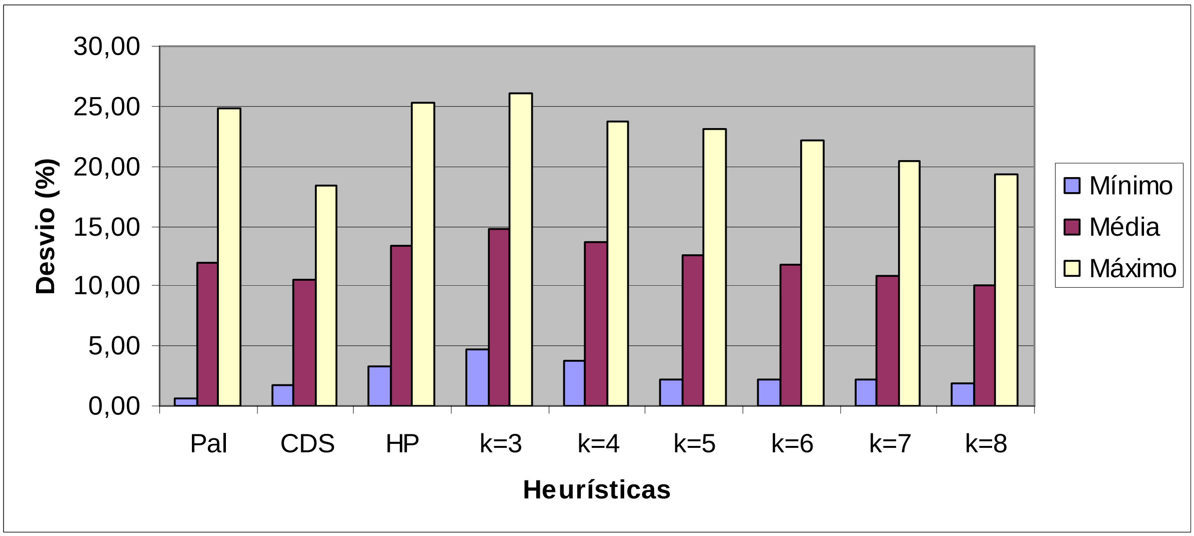

Figura 2 – Representação gráfica das soluções, com base na média, mínimo e máximo.

Algumas observações podem ser retiradas destes experimentos, tais como:

a) HPk, com k=8, apresentou o melhor desempenho entre todos os métodos avaliados, com desvio médio geral de 10,10 %, enquanto HPk, com k=3, obteve o pior desempenho médio 14,79 %;
b) PAL obteve o menor desvio (0,70 %) para um problema da classe (100, 05), onde no mesmo problema CDS e HP tiveram desvios de 5,17 % e 7,08 %, respectivamente, enquanto o melhor de HPk foi para k=6, com 3,19 %;

c) O melhor desempenho de HPk aconteceu na classe (020, 05) com desvio de 1,91 % e k=8, enquanto PAL e CDS obtiveram desvios de 5,89 % e 4,78 %, respectivamente, no mesmo problema;
d) O pior desempenho foi obtido por HPK, para k=3;
e) Pela Figura 2, vê-se que HPk tende a melhorar seus resultados quando k cresce de valor;
f) Todos os métodos, exceto HPk para k≥5, apresentaram comportamento idêntico em relação ao tempo computacional de execução, em média menos de 1 minuto.
g) O tempo computacional de HPk, para problemas de maior porte, também foi considerado aceitável, pois o tempo máximo foi de 936 segundos (15,6 minutos) para um problema da classe (100, 20), somente para k=8. Isto significa que o nosso método se adapta muito bem a problemas práticos. A Figura 3, mostra uma evolução do tempo computacional de HP e HPk, com o valor de k variando de 3 a 8.
h) O tempo de HPk cresce rapidamente para uma pequena variação no valor de k≥7, conforme Figura 3.

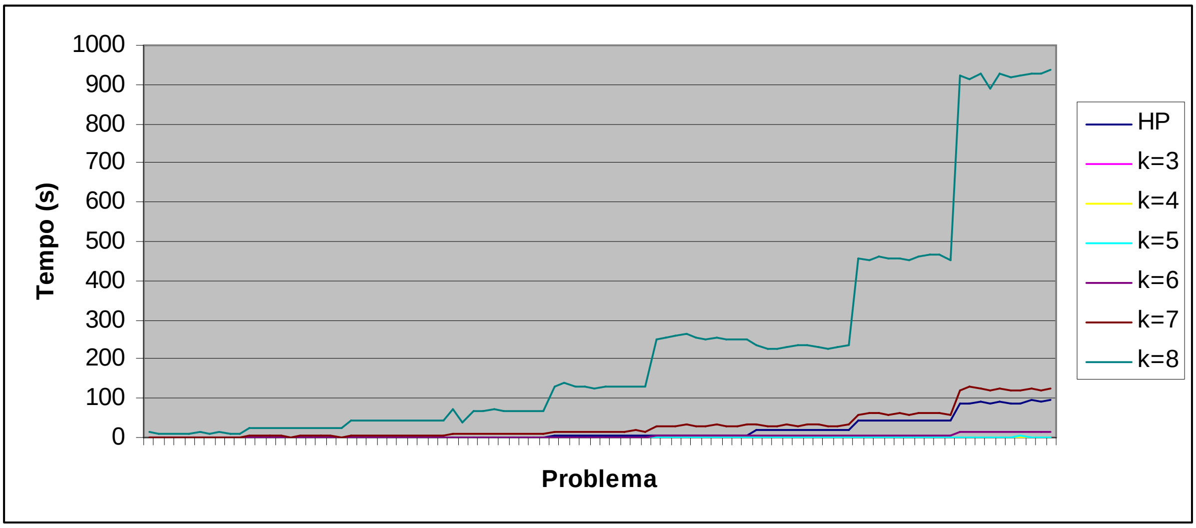

Figura 3 – Tempo de execução de HPk nos experimentos computacionais.

4. Conclusão

Uma maneira de reduzir a complexidade na resolução do problema computacionalmente é através do uso de heurísticas, que embora não garantam a solução exata, estabelece um compromisso entre os resultados obtidos e o custo computacional. Nossos experimentos computacionais, baseados em problemas da literatura, indicam que HPk obteve sucesso na resolução dos problemas propostos. A diversificação na busca tende a gerar bons resultados mesmo que a busca se torna exaustiva. Pode-se afirmar que HP e HPk são procedimentos adequados e baratos computacionalmente, exigem poucos recursos computacionais, para serem aplicados na resolução do problema da programação *Flow Shop* permutacional.

HPk pode ser aplicada a classe de problemas de otimização combinatória permutacional sem maiores dificuldades. Escolheu-se o FSP, como o representante dos POCP a ser estudado para uma aplicação da nossa técnica. Esta escolha não se deu ao acaso, visto que se trata de um dos problemas bastante estudado com instâncias já consolidadas na literatura. A modelagem de um problema de otimização combinatória permutacional, como um FSP, não prejudica em nada a generalidade com que ele poderia ser tratado aqui.

Trabalhos futuros podem ser realizados, tais como:

- Para diminuir o tempo de execução de HPk usaria-se o processamento paralelo ou distribuído, em vez de processamento seqüencial. Uma vez que no procedimento é

possível que a busca seja feita em paralela através dos diversos processadores, existentes e disponíveis para tal.
- Aplicar HPk em outros problemas da classe dos POCP.
- Estudar o valor adequado para o parâmetro *k* do procedimento, que não comprometa os recursos computacionais e gere boas soluções para o problema.

Agradecimentos
Os autores agradecem o apoio da Universidade Federal do Ceará e do CNPq (processo 302176/03-9).

5. Bibliografia

Beasley, J.E. OR-Library: Distributing Test Problems by Eletronic Mail. *Journal of the Operations Research Society*, 41: 1069-1072, 1990.

Campbell, H.G., Dudek, R.A., Smith, M.L. A heuristic algorithm for the n-job, m-machine sequencing problem. *Management Science*, 16: B630–B637, 1970.

Conway, R.W., Maxwell, W.L., Miller, L.W. *Theory of scheduling. Reading*, MA: Addison-Wesley; 1967.

Garey, M.R., Johnson, D.S., Sethi, R. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1(2):117–29, 1976.

Graham, R.L., Lawler, E.L., Lenstra, J.K., Kan, A.H.G.R. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

Gupta, J.N.D. A Functional Heuristic Algorithm for the Flow-Shop Scheduling Problem. *Operational Research Quarterly* 22, 39-47, 1971.

Gupta, J.N.D., Stafford Jr, E.F. Flowshop scheduling research after five decades. *European Journal of Operational Research* 169, 699–711, 2006.

Hundal, T.S. and Rajgopal, J. An Extension of Palmer's Heuristic for the Flow-Shop Scheduling Problem. *International Journal of Production Research* 26, 1119-1124, 1988.

Ignall, E. and Schrage, L.E. Application of Branch and Bound Technique to some Flow-Shop Problem. *Operations Research* 13, 400-412, 1965.

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

Moccellin, J. V. Um método Heurístico Híbrido Algoritmo Genético – Busca Tabu para a Programação Flow Shop Permutacional. Anais do XXI SBPO-Simpósio Brasileiro de Pesquisa Operacional, Juiz de Fora-MG, Brasil,

Nawaz, M., Enscore Jr., E.E. and Ham, I. A Heuristic Algorithm for the m-Machine n-Job Flow-Shop Sequencing Problem. *OMEGA* 11, 91-95, 1983.

Palmer, D.S. Sequencing jobs through a multistage process in the minimum total time - a quick method of obtaining a near optimum. *Operational Research Quartetly* 16: 101-107, 1965.

Potts, C.N. An Adaptive Branching Rule for the Permutation Flow-Shop Problem. *European Journal of Operational Research* 5, 19-25, 1980.

Reeves, C. R. *Modern Heuristic Techniques for Combinatorial Problems*. McGraw-Hill, London, 1995.

Ruiz, R., Maroto, C., Alcaraz, J. Two newrobust genetic algorithms for the flowshop scheduling problem. *The International Journal the Management Science* (Omega), 34 : 461 – 476, 2006.

Selen, W.J. and Hott, D.D. A Mixed-Integer Goal Programming Formulation of theStandard Flow-Shop Scheduling Problem. *Journal of the Operational Research Society* 37, 1121-1128, 1986.

Silva, J. L. C. e Soma, N. Y. Uma heurística para Problemas de Otimização Combinatória Permutacional. *Anais do XXXIII SBPO-Simpósio Brasileiro de Pesquisa Operacional*, Campos do Jordão-SP, Brasil, 2001.

Taillard, E. Benchmarks for Basic Scheduling Problems. *European Journal of Operational Research* 64, 278-285, 1993.

Whitley, D., Starkweather, T. and Shaner, D. *The traveling salesman and sequence scheduling: quality solutions using genetic edge recombination*. Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, 350-372, 1991.

Wilson, J.M. Alternative Formulations of a Flow Shop Scheduling Problem. *Journal of the Operational Research Society* 40, 395-399, 1989.