*J. King Saud Univ.,* Vol. 18, *Eng. Sci.* (1), pp. 83-99, Riyadh (1426H./2005)

# New Heuristics to Minimize the Mean Flow Time for Static Permutation Flowshop Problems

**Ibrahim M. Al-Harkan**
*Industrial Engineering Department, College of Engineering, King Saud University,*
*P.O. Box 800, Riyadh 11421, Saudi Arabia*
*email: imalhark@ksu.edu.sa*

(Received 23 February, 2005; accepted for publication 25 April, 2005)

**Abstarct.** This paper presents three new heuristic procedures in an attempt to improve the solution's quality of an earlier proposed heuristic in the literature. The heuristcis are developed to minimize average flow time in static and dynamic flexible manufacturing cell environment for n-job m-machine problem. The first heuristic is a modified version of Chan and Bedworth's heuristic (Inter. J. Prod. Res., Vol 28, pp. 2037-2049, 1990) which attempts to improve its solution's performance. The second heuristic is based on the development of an index function to select a job to be appended to a partial sequence. Finally, a simulated annealing algorithm is developed to be also used for testing and comparisons. The three heuristics along with that of Chan and Bedworth are tested by solving 180 randomly generated problems. Based on the results of this experemintal work, the improved Chan and Bedworth's heuristic was superior to the other three heuristics when minimizing the average flow time. With respect to the makespan as an objective function, the heuristic using the index function outperformed the remaining heuristics.

***Keywords***: Flowshop, Scheduling, Heuristics.

## 1. Introduction

In many manufacturing settings such as systems with flexible manufacturing cells, the most natural performance metric is the flow time since it measures the quality of service in terms of the time customers have to wait for their jobs to be processed. It is also a good indication for the cells utilization and the work-in-porcess level. In this context, the manufacturing manager would insist that the average time that customers experience to receive their jobs is not too high. This manager goal can be achieved by determining the processing sequence of the jobs that minimizes the average flow time.

In this paper, we limit our work to the study of static permutation flowshop sequencing problems with the average flow time (or equivalently, total completion time) as a performance criterion. We consider the situation where $n$ jobs are available at time zero for processing and are processed on $m$ machines in the same technological order. Moreover, the jobs are processed on every machine in the same order with no preemption. The processing times, $P_{ij}$ ($i$=1, 2, …, $n$ and $j$ = 1, 2, …, $m$) of every job on each machine are deterministic.

In recent years, the focus of the scheduling research community has been on the development of heuristic procedures for some well known NP-hard scheduling problems such as the one considered in this paper. In fact, given that the average flow time minimization problem was shown to be NP-hard (Gonzalez and Sahni [1]) for $m \geq 2$, it is necessary to develop computationally efficient heuristic procedures that generate close optimal sequences.

From a review of the literature, it can be noticed that several heuristic approaches in the field of flowshop scheduling have been designed to minimize both the maximum flow time and the makespan. Some of these approchaes are given by Palmer [2], Ashour [3], Campbell, Dudek and Smith [4], Gupta [5], Baker [6], Dannenbring [7], King and Spachis [8], Stinson and Smith [9], Nawaz *et al.* [10], Hundal and Rajgopal [11], Osman and Potts [12], Widmer and Hertz [13], Taillard [14], Ogbu and Smith [15], Ho and Chang [16], Sarin and Lefoka [17], Werner [18], Reeves [19], Chen *et al.* [20], Reeves [21], Ishibuchi *et al.* [22], Zegordi *et al.* [23], Moccellin [24], Murata *et al.* [25], Nowicki and Smutnicki [26], Koulamas [27], Stützle [28], Ben-Daya and Al-Fawzan [29], Reeves and Yamada [30], Suliman [31], Moccellin and dos Santos [32], Ponnambalam *et al.* [33], Davoud Pour [34], Allahverdi and Aldowaisan [35], Framinan *et al.* [36], and Wang and Zheng [37],… etc. However, relatively few papers exist for other performance measures such as the average flow time. Chan and Bedworth [38] proposed a heuristic for minimizing average flow time for static and dynamic flexible manufacturing cells. Their paper deals with cell scheduling when minimizing the mean flow time for n-job and m-machine problem in static and dynamic environments.

The purpose of this paper is first to develop a heuristic which attempts to improve the quality of solution obtained by Chan and Bedworth's heuristic. Then, another heuristic procedure is proposed which is based on an index function to sequence the jobs. Finally, a simulated annealing algorithm was developed to be also used for testing and comparisons.

This paper is organized as follows. The following section describes Chan and Bedworth's heuristic. In section 3, the improved Chan and Bedworth's heuristic is presented. In section 4, the index function heuristic is discussed. The simulated annealing algorithm is introduced in section 6. In section 7, the four heuristics are tested and compared. Finally, in section 8, some conclusions and future recommendations are given.

## 2. Chan and Bedworth's Heuristic

The purpose of Chan and Bedworth 's work is to minimize the average flow time or other performance measures (e.g. machine utilization, mean throughput, and the makespan) for n-job and m-machine problem in static and dynamic environments. Chan and Bedworth argue that the mean flow time is a logical criterion in an automated cell because it will represent an ongoing streams of parts. The cell scheduling problem for an automated manufacturing cell can be classified as follows:

1) A static cell scheduling problem which determines the job sequence when a group of jobs arrive at a cell, usually on a standard-fixture pallet, and
2) A dynamic cell scheduling problem which determines the job sequence when the initial machine in the cell becomes available.

Chan and Bedworth's heuristic is based on the comparison of the mean flow time for all possible pair of jobs. For two-job, *m*-machine problem, two sequences are evaluated based on the mean flow time, and accordingly one job should proceed the other. Their heuristic procedure can be summarized in an algorithmic form as follows:

**Algorithm**

Step 1: Retrieve the processing times from the cell database.

Step 2: Compute temporary, simplified flow times for each pair of jobs using the following equation:

$$\mathbf{F_{m(ij)} = 2P_{i1} + \sum_{k=2}^{m-1} P_{ik} + R^{*}_{m}}$$

where: $R_1 = 0$

$R_2 = P_{j2} + \max(P_{j1}, P_{i2})$

$R_3 = P_{j3} + \max(R_2, P_{i2} + P_{i3})$

$R_4 = P_{j4} + \max(R_3, P_{i2} + P_{i3} + P_{i4})$

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

$R^{*}_{m} = R_m - P_{jm} \quad m \geq 2$

for example $F_{m(12)}$, $F_{m(21)}$, $F_{m(13)}$, $F_{m(31)}$, $F_{m(23)}$, and $F_{m(32)}$ are six temporary flow times for a three-job, m-machine problem.

Step 3: Compare each pair of temporary flow times (i.e. $F_{m(12)}$, $F_{m(21)}$ is one such pair) and pick the smallest; assign an asterisk to the starting job for that pair. Perform this step for all pairs of flow times. Count the number of asterisks assigned to each job when all pairs have been evaluated and sequence that jobs with the largest number of asterisks first, next largest number of asterisks second, and so on.

The following example from Chan and Bedworth 's paper illustrates how the heuristic works. Table 1 shows the job data for four-job four-machine problem.

**Table 1. Processing time for 4 jobs on 4 machines**

| Job No. | M1 | M2 | M3 | M4 |
|---|---|---|---|---|
| 1 | 22 | 11 | 19 | 21 |
| 2 | 9 | 14 | 16 | 2 |
| 3 | 20 | 19 | 4 | 2 |
| 4 | 10 | 18 | 6 | 7 |

Since there are four jobs, then the total number of temporary flow times is twelve. For the purpose of illustration, the first two pairs are calculated (i.e. $F_{4(12)}$, and $F_{4(21)}$ ). The calculation for the simplified flow time is as follows:

$$F_{4(12)} = (2)(22) + (11 + 19) + 51 = 125$$

The first term is $P_{11}$ and the second term is $P_{12} + P_{13}$. The last term is calculated as follows:

$$R_1 = 0$$
$$R_2 = 14 + \max(9, 11) = 25$$
$$R_3 = 16 + \max(25, 30 ) = 46$$
$$R_4 = 2 + \max(46, 51) = 53$$
$$R^*_m = 53 -2 = 51.$$

For the sequence 21 we have

$$F_{4(21)} = (2)(9) + (14 + 16) + 52 = 100,$$

where the last term is calculated as follows:

$$R_1 = 0$$
$$R_2 = 11 + \max(22, 14) = 33$$
$$R_3 = 19 + \max(33, 30 ) = 52$$
$$R_4 = 21 + \max(52, 32) = 73$$
$$R^*_m = 73 -21 = 52.$$

Table 2 shows all temporary flow time values for the 12 pairs.

**Table 2. Pair of sequences and corresponding temporary flow times**

| Sequence pair | P* values |
|---|---|
| 1-2:2-1 | 125:100 |
| 1-3:3-1 | 125:115 |
| 1-4:4-1 | 125:96 |
| 2-3:3-2 | 91:112 |
| 2-4:4-2 | 86:92 |
| 3-4:4-3 | 106:87 |

Table 3 shows the best sequence in pair. From this table, it is clear that job 2 has 3 asterisks, job 4 has 2 asterisks, job 3 has 1 asterisk, and job 1 has none. Therefore, according to the sum of asterisks the best sequence is 2-4-3-1 which has a mean flow time of 68.25 (which is optimum).

**Table 3. The best pair sequence**

| Sequence pair | The best sequence in pair |
|---|---|
| 1-2:2-1 | 2*-1 |
| 1-3:3-1 | 3*-1 |
| 1-4:4-1 | 4*-1 |
| 2-3:3-2 | 2*-3 |
| 2-4:4-2 | 2*-4 |
| 3-4:4-3 | 4*-3 |

To illustrate further the heuristic procedure, another example is solved with 6 jobs and 3 machines. Table 4 shows the job data.

**Table 4. Processing time for 6 jobs on 3 machines**

| Job no. | M1 | M2 | M3 |
|---|---|---|---|
| 1 | 5 | 8 | 20 |
| 2 | 6 | 30 | 6 |
| 3 | 30 | 4 | 5 |
| 4 | 2 | 5 | 3 |
| 5 | 3 | 10 | 4 |
| 6 | 4 | 1 | 4 |

Since there are 6 jobs, then there are a total of 30 temporary flow times. Table 5 shows all temporary flow time values for the 30 pairs.

**Table 5. Pair of sequences and corresponding temporary flow times**

| Sequence pair | P* values |
|---|---|
| 1-2:2-1 | 56:80 |
| 1-3:3-1 | 52:77 |
| 1-4:4-1 | 46:17 |
| 1-5:5-1 | 46:34 |
| 1-6:6-1 | 46:22 |
| 2-3:3-2 | 78:100 |
| 2-4:4-2 | 78:45 |
| 2-5:5-2 | 82:56 |
| 2-6:6-2 | 78:45 |
| 3-4:4-3 | 73:43 |
| 3-5:5-3 | 78:50 |
| 3-6:6-3 | 73:43 |
| 4-5:5-4 | 24:31 |
| 4-6:6-4 | 17:16 |
| 5-6:6-5 | 30:22 |

Table 6 shows the best sequence in pairs. From this table, it is clear that job 6 has 5 asterisks, job 4 has 4 asterisks, job 5 has 3 asterisks, job 1 has 2 asterisks, job 2 has one asterisk, and job 3 has none. Therefore, according to the sum of asterisks the best sequence is 6-4-5-1-2-3 which has a mean flow time of 38.67.

**Table 6. The best pair sequence**

| Sequence pair | The best sequence in pair |
|---|---|
| 1-2:2-1 | 1*-2 |
| 1-3:3-1 | 1*-3 |
| 1-4:4-1 | 4*-1 |
| 1-5:5-1 | 5*-1 |
| 1-6:6-1 | 6*-1 |
| 2-3:3-2 | 2*-3 |
| 2-4:4-2 | 4*-2 |
| 2-5:5-2 | 5*-2 |
| 2-6:6-2 | 6*-2 |
| 3-4:4-3 | 4*-3 |
| 3-5:5-3 | 5*-3 |
| 3-6:6-3 | 6*-3 |
| 4-5:5-4 | 4*-5 |
| 4-6:6-4 | 6*-4 |
| 5-6:6-5 | 6*-5 |

## 3. Improved Chan and Bedworth's Heuristic

This section describes an improvement step which can be added to the heuristic proposed by Chan and Bedworth to enhance its performance. In step 3 of the Chan and Bedworth's heuristic, there are two possibilities of improvement, which are raised by the following questions:

1) What would happen if the simplified flow times for a pair of jobs are the same in both sequences?
2) What would happen if more than one job have the same number of asterisks?

Therefore, the two possibilities of improvement are:

1) When the simplified flow time for a pair of jobs is the same then give asterisks to both jobs.
2) When the sums of asterisk for two jobs or more are the same then form all possible sequences and select the best schedule. For example, assume that there are five jobs (i.e. 1, 2, 3, 4, 5) and the sums of asterisks for the first two jobs (i.e. 1 and 2) in the sequence are equal, then two sequences 1-2-3-4-5 and 2-1-3-4-5 should be considered. Next, calculate the average flow time for the two sequences and select the sequence with the minimum average flow time.

The two possibilities of improvement are added to the proposed heuristic by Chan and Bedworth. The following numerical example is used to illustrate the proposed improvement step. Assume that there are 4-job 3-machine problem where the job data are given in Table 7.

**Table 7. Processing time for 4 jobs on 3 machines**

| Job no. | M1 | M3 | M4 |
|---|---|---|---|
| 1 | 1 | 8 | 4 |
| 2 | 2 | 4 | 5 |
| 3 | 6 | 2 | 8 |
| 4 | 3 | 9 | 2 |

From the above data, all of the simplified flow times are calculated and are given in Table 8.

**Table 8. Pair of sequences and corresponding temporary flow times**

| Sequence pair | P* values |
|---|---|
| 1-2:2-1 | 22:20 |
| 1-3:3-1 | 22:24 |
| 1-4:4-1 | 27:32 |
| 2-3:3-2 | 17:24 |
| 2-4:4-2 | 21:28 |
| 3-4:4-3 | 26:26 |

Table 9 shows the best sequence in pair.

**Table 9. The best pair sequence**

| Sequence pair | The best sequence in pair |
|---|---|
| 1-2:2-1 | 2*-1 |
| 1-3:3-1 | 1*-3 |
| 1-4:4-1 | 1*-4 |
| 2-3:3-2 | 2*-3 |
| 2-4:4-2 | 2*-4 |
| 3-4:4-3 | 4*-3 and 3*-4 |

From Table 9, it is clear that job 2 has 3 asterisks, job 1 has 2 asterisks, job 3 has 1 asterisk, and job 4 has 1 asterisk. Therefore, according to the sum of asterisks there are two sequences, the first sequence is 2-1-4-3 which has average flow time of 21.75 and the second is 2-1-3-4 which has average flow time of 20.75. Thus, the second sequence is better than the first, therefore it is selected.

## 4. Index Function Heuristic

In this section, a more complicated procedure is proposed to minimize the average flow time. This heuristic is based on an index function to be used for determining the position of each job in the sequence. The sequence is generated by appending jobs one by one to a partial sequence developed in previous steps of the heuristic. The index function represents the total machine idle times and total waiting times for a job $u = 1, 2, \ldots, n$ which is to be processed after a job $v \neq u = 1, 2, \ldots, m$ and assuming that this pair of jobs are the only jobs to be processed on the $m$ machines. More formally, this index function is defined as follows:
D(v,u): index assigned to two adjacent jobs v and u where job v precedes job u. The D(v,u) is calculated for a pair of job as follows: $\mathbf{D(v,u)} = \sum_{\mathbf{j=1}}^{\mathbf{m-1}} \left| \mathbf{P_{v,j+1}} - \mathbf{P_{u,j}} \right|$, where $v \neq u$, v = 1, ..., n, and u = 1,...n. For n jobs, there are n(n-1) indices. For example, assume that there are three 3 jobs then there will be six pairs of indices to be computed. These pairs are: D(1,2) D(1,3), D(2,1), D(2,3), D(3,1), and D(3,2). Note that when $P_{v,j+1} > P_{u,j}$, then job $u$ has to wait between machines $j$ and $j+1$ before it can be processed. On the other hand when $P_{v,j+1} < P_{u,j}$, then machine $j$+1 is kept idle until job $u$ is compltely processed on machine $j$.

Next, a discussion of the different steps of the heuristic is presented. Prior to describing the different steps, some terms that are used in the heuristic need to be interpreted. Calculate all D(v,u) implies using the D(v,u) formula presented above. Scheduling a job means schedule it at $L^{th}$ position in the processing sequence. Index H is a counter for the total number of jobs which have been scheduled (i.e. at the end of the heuristic, H = n). Index L indicates the position of a job in the processing sequence. These different steps of the heuristic are outlined as follows:

Step 1: Set H = 0, and L= 0.
H: index for the total number of jobs that have been scheduled.
L: index for the job scheduled at $L^{th}$ position in the processing sequence.
Step 2: Calculate all D(v,u)s by repeating this step n(n-1) times, where v = 1,2,...,n and u = 1,2,...,n where v ≠ u for any D(v,u).
Step 3: Find the minimum D(v,u), then find the next four minimum D(v,u)s over v and u, where v = 1,...,n-H, u = 1,...,n-H, v ∈ U, and u ∈ U. Where U is the set of (n-H) jobs as yet to be scheduled. If there are more than one equal D(v,u), then all the corresponding pairs (v,u) are considered as candidates. Moreover, when determining the next four minimum D(v,u), consider all pairs (v, u) in case of equal D(v,u)
Step 4: For all the (v,u)'s found in step 3 consider them as starting partial sequeces. Set S = number of sequences.
Step 5: Repeat steps 6 and 7 *S* times.
Step 6: Set L = 2 and H = H + 2 then;
Schedule the first two jobs in positions $(L-1)^{th}$ and $L^{th}$.
Step 7: (a) Find the next minimum D(v,u) over u, where u = 1,...,n-H and u ∈ U, where U is the set of (n-H) jobs as yet to be scheduled.
If D(v,u) is unique proceed to step 7(b), otherwise increase S by one and retrieve all the jobs assigned to this partial schedule and consider them as new schedule with pair that is equal to this D(v,u), then proceed to step 7 (b).
(b) Set L = L + 1
Set H = H + 1
Schedule job selected on step 7(a) at $L^{th}$ position, If H = n STOP, otherwise go to step 7 (a).
Step 8: Calculate the average flow times for all schedules generated and select the schedule with the lowest average flow time.

The heuristic at step 8 is terminated. The proposed heuristic is based on an efficient idea since it gives the chance to start the sequence using different starting partial sequences. This in turns allows the search to go through different paths and different neighborhood of job sequences.

Consider the same example in the improved heuristic section (see Table 7) which can be used to illustrate the different steps of the heuristic. In step 2 we calculate all the D(v,u)s. The following is the D(v,u) matrix values.

**Table 10. Matrix of D(v, u) values**

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | - | 6 | 4 | 10 |
| 2 | 6 | - | 5 | 5 |
| 3 | 1 | 4 | - | 2 |
| 4 | 14 | 9 | 3 | - |

From the above matrix, the minimum D(v,u) is for the pair (3,1), and the next four minimum D(v,u)'s are obtained for the pairs (3,4), (4,3), (3,2), (1,3), (2,3), and (2,4). For the partial sequence (3,1), the next job to be appended is job 2 since D(1,2) is smaller than D(1,4). Therefore, the complete sequence starting with pair (3,1) is 3-4-2-1 with a mean flow time of 33. For the remaining pairs we have the following sequences: 3-4-2-1 with $\overline{F}$ = 34, 4-3-1-2 with $\overline{F}$ = 23.25, 3-2-4-1 with $\overline{F}$ = 23.25, 1-3-4-2 with $\overline{F}$ = 21.25, 2-4-3-1 with $\overline{F}$ = 20.5, and 2-3-1-4 with $\overline{F}$ = 20.5. Therefore, the last two schedules are the best, so that one of these two schedules can be selected.

## 5. The Simulated Annealing Algorithm

In this section, the simulated annealing approach proposed by Burkard *et al.* [39] is adopted to be used in minimizing the average flow time for the flowshop scheduling problem. The different steps of the adopted procedure are as follows:

Step 1: Set Restart = 10, number of repetition for the procedure.
Step 2: Initialize all variables.
Step 3: Generate a random schedule, call it S, then transform S into S'
Step 4: Set $t$ = 100000, $a$= 0.5, Trial = $n^2$, and Change = 1, where $t$ represents the mobility parameter and $a$ is a positive factor less than one by which $t$ is reduced after each trial cycle. According to the number of jobs, the value of Trail is set and it is considered to be a large number.
Step 5: Do while Change = 1
    *Comment*: as long as changes occur in the objective function value, change is set to 1. The procedure stops if change = 0 during a complete cycle of random trails.
    Change = 0.
    Rep = 1 (Rep is counter for Trail number)
    Do while Rep < Trial
        Select two jobs randomly (uniformly distributed (1,n)) in S' and swap them.
        Calculate the completion times for the machines in both schedules S and S', then calculate the average flow time for both schedules.
        Set MFT(S) = Mean Flow Time for S.
        Set MFT(S') = Mean Flow Time for S'.
        Set Delta = MFT(S') - MFT(S).

If Delta < 0 then go to Accept, otherwise continue.
PDelta = $e^{(-Delta/t)}$.
Generate random variable x uniformly distributed in (0,1).
If x < PDelta then go to Accept else go to Exit.
Accept: S = S', and MFT(S) = MFT(S')
If Delta ≠ 0 then Change = 1
Exit: Rep = Rep + 1
LOOP.
Trial = Rep * 1.1 (Rep is multiplied after each cycle by 1.1, such that as t decreases, more trails are performed at constant t ).
t = t * a (a is the factor reducing mobility parameter t after each cycle).
LOOP.
Step 6: Retain the schedule obtained by the above procedures and,
If Restart = 10 then select the best schedule and STOP, otherwise go to step 2.

## 6. Experimental Comparison of the Heuristics

To assess the quality of solutions obtained by the four heuristics, they were coded in QUICK BASIC. Then, different computational experiments were performed using randomly generated data. These experimental tests were generated according to the following parameters: three levels for the number of machines (m) = 3, 4, or 5, three levels for the number of jobs (n) = 15, 20, or 25, and the processing times ($P_{ij}$) were triangularly distributed with a minimum value of 3, a maximum value of 20, and a mode value of 10. Thus, there were three cases for m, three cases for n, and one case for $P_{ij}$, a total of nine combinations of parameters. Twenty random problems were generated for each combination. For the 180 problems, the tests were performed as follows:

1) The process times for the jobs were generated;
2) The computer codes for the four heuristics were run;
3) The average flow time and the makespan were recorded;
4) The percentage errors were calculated.

The percentage error measure was used to test the quality of solutions obtained by each of the four heuristics. The percentage of error ($\alpha$) is defined in terms of the performance measure used in this research, (i.e., the average flow time). The percentage error was calculated for each heuristic as follows:

$$\alpha_k = \frac{\overline{F}_k - \overline{F}_{\min}}{\overline{F}_{\min}} * 100$$

where the index $k$ = 1 for Chan and Bedworth's heuristic, 2 for the improved Chan and Bedworth's heuristic, 3 for the index function heuristic, and 4 for the simulated annealing algorithm. The $\overline{F}_{min}$ is the minimum average flow time obtained by one of the four heuristics for a specific scheduling problem.

The results for each of the nine combinations obtained by each of the four heuristics are summarized in Table 11. In Table 11, each row represents the averages obtained for the twenty problems for each combination.

Also, from Table 11 we can construct the following two tables to rank and sort the four heuristics by giving them weight according to their rank in each combination. The weights will help us to decide which of the heuristics is the best.

From Table 12, the sum of weights for each heuristic is as follows: for Chan and Bedworth's heuristic the sum of weight is 66, for the improved Chan and Bedworth's heuristic the sum of weights is 89, for index function heuristic the sum of weights is 65, and for the annealing heuristic the sum of weights is 62. Therefore, according to these weights the four heuristics can be ranked as follows: improved Chan and Bedworth's heuristic - Chan and Bedworth's heuristic - index function heuristic - annealing. However, it should be clear and fair to be mentioned that the difference in the quality of solutions obtained by Chan and Bedworth, index function, and annealing heuristics is not significant.

From Table 13, the sum of weights for each heuristic is as follows: for Chan and Bedworth's heuristic the sum of weight is 65, for the improved Chan and Bedworth's heuristic the sum of weight is 73, for index function heuristic the sum of weight is 87, and for the annealing the sum of weight is 81. Therefore, according to these weight we can rank the heuristics as follows: Index function-annealing-improved-original Chan and Bedworth. It should be clear that none of the heuristics had the same performace for the makespan and mean flow time.

## 7. Conclusion and Recommendations

The following conclusions are based on the findings from the experimental works based on the performance measure addressed in this study, namely the average flow time. Also, based on the makespan which was computed during the experimental work.

The results showed that for the average flow time measure, the improved Chan and Bedworth's heuristic was the best among the four heuristics. Also, both Chan and Bedworth and index function heuristics performed better than the simulated annealing algorihtm.

For the makespan computed, the index function heuristic outperformed the remaining heuristics. Also, the simulated annealing algorithm performed better than Chan and Bedworth, and improved heuristics.

**Table 12. Ranks of the four heuristics according to the average flow time measure**

| Combination no. | First in rank (w*=10) | Second in rank (w=9) | Third in rank (w=8) | Fourth in rank (w=7) |
|---|---|---|---|---|
| 1 | 2 | 4 | 3 | 1 |
| 2 | 2 | 4 | 3 | 1 |
| 3 | 2 | 1 | 3 | 4 |
| 4 | 2 | 4 | 3 | 1 |
| 5 | 3 | 2 | 1 | 4 |
| 6 | 2 | 1 | 3 | 4 |
| 7 | 2 | 4 | 1 | 3 |
| 8 | 2 | 1 | 3 | 4 |
| 9 | 2 | 1 | 3 | 4 |

* w is the weight given to each heuristic according to its rank

**Table 13. Ranks of the four heuristics according to the makespan measure**

| Combination no. | First in rank (w=10) | Second in rank (w=9) | Third in rank (w=8) | Fourth in rank (w=7) |
|---|---|---|---|---|
| 1 | 4 | 2 | 3 | 1 |
| 2 | 3 | 4 | 2 | 1 |
| 3 | 4 | 3 | 1 | 2 |
| 4 | 3 | 4 | 2 | 1 |
| 5 | 3 | 4 | 1 | 2 |
| 6 | 3 | 2 | 4 | 1 |
| 7 | 3 | 4 | 2 | 1 |
| 8 | 3 | 4 | 2 | 1 |
| 9 | 3 | 2 | 4 | 1 |

In summary, the improved Chan and Bedworth's heuristic is an effective method of scheduling n jobs on m machines when minimizing the average flow time and index function heuristic is an effective method of scheduling n jobs on m machines when minimizing the makespan.

An attempt to improve the sequence of jobs obtained by the annealing approach can be performed as follows: instead of selecting a random schedule as a seed we might start with one of the solutions obtained by the other three heuristics.

***Acknowledgements.*** The author wishes to thank the anonymous referees for their valuable suggestion and comments.

## References


[1] Gonzalez, T. and Sahni, S. “Flow Shop and Job Shop Schedules.” *Operation Research*, 26 (1978), 36-53.

[2] Palmer, D. ”Sequencing Jobs Through a Multi-stage Process in the Minimum Total Time: A Quick Method of Obtaining a Near Optimum.” *Operational Research Quarterly,* 16, No. 1, (1965), 101–107.

[3] Ashour, S. "An Experimental Investigation and Comparative Evaluation of Flow Shop Scheduling Techniques." *Operation Research*, 18 (1970), 541-548.

[4] Campbell, H. G., Dudek, R. A. and Smith, M. L. “A Heuristic Algorithm for the n-Job, m-Machine Sequencing Problem.” *Management Science,* 16, No. 10 (1970), B630-B637.

[5] Gupta, J.N.D. "A Heuristic Algorithm for Multistage Flow Shop Scheduling Problem." *AIII Transactions*, 4, No. 1 (1972), 11-18.

[6] Baker, K. R. "A Comparative Survey of Flow Shop Algorithms." *Operation Research*, 23 (1974), 62-73.

[7] Dannenbring, D. G. “An Evaluation of Flow Shop Sequencing Heuristics.” *Management Science,* 23, No. 11 (1977), 1174-1182.

[8] King, J.R. and Spachis, A.S. “Heuristics for Flow Shop Scheduling." *International Journal of Production Research*, 18, No. 3 (1980), 345-357.

[9] Stinson, J.P. and Smith, A.W. "A Heuristic Programming Procedure for Sequencing the Static Flow Shop.” *International Journal of Production Research*, 20, No. 6 (1980), 753-764.

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

[11] Hundal, T.S. and Rajgopal, J. "An Extension of Palmer's Heuristic for the Flow Shop Scheduling Problem.” *International Journal of Production Research*, 26, No. 6 (1988), 1119-1124.

[12] Osman, I. and Potts, C. "Simulated Annealing for Permutation Flow Shop Scheduling.” *OMEGA, The International Journal of Management Science*, 17, No. 6 (1989), 551-557.

[13] Widmer, M. and Hertz, A. “A New Method for the Flow Shop Sequencing Problem.” *European Journal of Operation Research*, 41 (1989), 186-193.

[14] Taillard, E. “Some Efficient Heuristic Methods for Flow Shop Sequencing Problem.” *European Journal of Operation Research*, 47 (1990), 65-74.

[15] Ogbu, F. and Smith, D. “Simulated Annealing for the Permutation Flow Shop Problem.” *OMEGA, The International Journal of Management Science*, 19, No. 1 (1990), 64-67.

[16] Ho, J.C. and Chang, Y. L. “A New Heuristic for the n-Job, m-Machine Flow Shop Problem.” *European Journal of Operational Research,* 52 (1991), 194–202.

[17] Sarin, S. and Lefoka, M. “Scheduling Heuristic for the n-Job m-Machine Flow Shop. *OMEGA, The International Journal of Management Science,* 21, No. 2 (1993), 229–234.

[18] Werner, F. “On the Heuristic Solution of the Permutation Flow Shop Problem by Path Algorithms.” *Computers and Operations Research,* 20, No. 7 (1993), 707–722.

[19] Reeves, C.R. “Improving the Efficiency of Tabu Search for Machine Scheduling Problems.” *Journal of the Operational Research Society,* 44, No. 4 (1993), 375–382.

[20] Chen, C.L., Vempati, V.S. and Aljaber, N. “An Application of Genetic Algorithms for Flow Shop Problems.” *European Journal of Operational Research,* 80 (1995), 389–396.

[21] Reeves, C.R. “A Genetic Algorithm for Flow Shop Sequencing.” *Computers and Operations Research,* 22, No. 1 (1995), 5–13.

[22] Ishibuchi, H., Misaki, S. and Tanaka, H. “Modified Simulated Annealing Algorithms for the Flow Shop Sequencing Problem.” *European Journal of Operational Research,* 81 (1995), 388–398.

[23] Zegordi, S.H., Itoh, K. and Enkawa, T. “Minimizing Makespan for Flow Shop Scheduling by Combining Simulated Annealing with Sequencing Knowledge.” *European Journal of Operational Research,* 85 (1995), 515–531.

[24] Moccellin, J.A.V. “A New Heuristic Method for the Permutation Flow Shop Scheduling Problem.” *Journal of the Operational Research Society,* 46 (1995), 883–886.

[25] Murata, T., Ishibuchi, H. and Tanaka, H. “Genetic Algorithms for Flow Shop Scheduling Problems.” *Computers and Industrial Engineering,* 30, No. 4 (1996), 1061–1071.

[26] Nowicki, E. and Smutnicki, C. “A Fast Tabu Search Algorithm for the Permutation Flow Shop Problem.” *European Journal of Operational Research,* 91 (1996), 160–175.

[27] Koulamas, C. “A New Constructive Heuristic for the Flow Shop Scheduling Problem.” *European Journal of Operational Research Society,* 105 (1998), 66–71.

[28] Stützle, T. “Applying Iterated Local Search to the Permutation Flow Shop Problem.” Technical Report, AIDA-98-04, FG Intellektik,TU Darmstadt, 1998.

[29] Ben-Daya, M. and Al-Fawzan, M. “A Tabu Search Approach for the Flow Shop Scheduling Problem.” *European Journal of Operational Research,* 109 (1998), 88–95.

[30] Reeves, C. and Yamada, T. “Genetic Algorithms, Path Relinking, and the Flow Shop Sequencing Problem.” *Evolutionary Computation,* 6, No. 1 (1998), 45–60.

[31] Suliman, S. “A Two-phase Heuristic Approach to the Permutation Flow Shop Scheduling Problem.” *International Journal of Production Economics,* 64 (2000), 143–152.

[32] Moccellin, J.A.V. and dos Santos, M.O. “An Adaptive Hybrid Meta-heuristic for Permutation Flow Shop Scheduling.” *Control and Cybernetics,* 29, No. 3 (2000), 761–771.

[33] Ponnambalam, S.G., Aravindan, P. and Chandrasekaran, S. “Constructive and Improvement Flow Shop Scheduling Heuristics: An Extensive Evaluation.” *Production Planning and Control,* 12, No. 4 (2001), 335–344.

[34] Davoud Pour, H. “A New Heuristic for the n-Job, m-Machine Flow Shop Problem.” *Production Planning and Control,* 12, No. 7 (2001), 648–653.

[35] Allahverdi, A. and Aldowaisan, T. “New Heuristics to Minimize Total Completion Time in m-Machine Flow Shops.” *International Journal of Production Economics,* 7 (2002), 71–83.

[36] Framinan, J.M., Leisten, R. and Rajendran, C. “Different Initial Sequences for the Heuristic of Nawaz, Enscore and Ham to Minimize Makespan, Idle Time or Flow Time in the Static Permutation Flow Shop Sequencing Problem.” *International Journal of Production Research,* 41, No. 1 (2003), 121–148.

[37] Wang, L. and Zheng, D.Z. “An Effective Hybrid Heuristic for Flow Shop Scheduling.” *The International Journal of Advanced Manufacturing Technology,* 21 (2003), 38–44.

[38] Chan, D. and Bedworth, D. "Design of a Scheduling System for Flexible Manufacturing Cells." ***Int. J. Prod. Res.***, 28, No. 11 (1990), 2037-2049.

[39] Burkard, R. E. and Rendl, F. "A Thermodynamically Motivated Simulation Procedure for Combinatorial Optimization Problems." ***Eur. J. of Opns. Res.***, 17 (1984), 169-174.

# خوارزميات جديدة لتقليل معدل الوفت الذي ضيه الشغل مائل خطوط الإنتاج الاستاتيكية ا تتابعة

**إبراهيم بن مد ا ركان**

*قسم الهندسة الصناعية، كلية الهندسة، جامعة الملك سعود*

*ص.ب: ٨٠٠ الرياض ١١٤٢١، المملكة العربية السعودية*

*البريد الالكتروني: imalhark@ksu.edu.sa*

(قدم للنشر في ٢٠٠٥/٢/٢٣م، وقبل للنشر في ٢٠٠٥/٤/٢٥م)

**ملخص البحث.** يقدم هذا البحث ثلاث طرق خوارزمية جديدة لتحسين الحل المستنبط من الخوارزمية المطورة من قبل شأن وبدروث وذلك لتقليل معدل الوقت الذي يمضيه الشغل في خطوط الإنتاج الديناميكية والاستاتيكية. الخوارزمية الأولى المطورة في البحث تعتمد في طريقة حلها على الخوارزمية المطورة من قبل شأن وبدروث ولذلك تعتبر خوارزمية محدثة لطريقة شأن وبدروث. الخوارزمية الثانية تعتمد في طريقة حلها على دلالة لاختيار الشغل خلال فترة البناء للحل. الخوارزمية الثالثة تعتمد في طريقة حلها على خوارزمية محاكاة الطرق. لتحديد مدى جودة الحل المستنبط من الخوارزميات المذكورة، تم بناء ١٨٠مسئلة عشوائياً ومن ثم تم حلها بالخوارزميات الأربع. ومن النتائج يتضح أن الخوارزمية الأولى المطورة في هذا البحث تعطي أقل قيمة لمعدل الوقت الذي يمضيه الشغل في خطوط الإنتاج، بينما الخوارزمية الثانية المطورة في هذا البحث تعطي أقل قيمة للوقت الذي تمضيه جميع الأعمال في خطوط الإنتاج.