DESIGNING A TABU SEARCH ALGORITHM TO MINIMIZE TOTAL FLOW TIME IN A FLOW SHOP

Jatinder N.D. Gupta*
Department of Management
Ball State University
Muncie, IN 47306, USA

and

Chuen-Lung Chen
Lee Yee Yap
Harshwardhan Deshmukh
Department of Industrial Engineering
Mississippi State University
Mississippi State, MS 39762,USA

الخلاصة :

تناقش هذه الورقة عملية تصميم طريقة حل متكاملة (تابو) تعتمد على طريقة ايضاحية لتصغير الزمن الكليّ في ورشة تدفق انسيابي تبادلية. حيث صُممت تجربة متعددة العوامل لدراسة تأثير العوامل المختلفة بطريقة منظمة على أداء طريقة الحل، وهذه العوامل هي: الحلّ الأول ونوع الحركة وحجم الجيرة ومقاس قائمة التابو وشرط التوقف ومستوى الطموح. واستخدمت طرق منحيات التطور وجداول التأثير ورسوم التأثير في تحديد أفضل تركيبة للعوامل أعلاه. ونقدم في هذه الورقة مناقشةً لنتائج المقارنة بين الطريقة المقترحة وأفضل الطرق المعروفة لحل هذه المسألة.

ABSTRACT

This paper discusses the process of designing a tabu search based heuristic for minimizing total flow time in a permutation flow shop. A factorial experiment is designed to systematically analyze the effects of various factors (namely, the initial solution, type of move, neighborhood size, tabu list size, stopping condition, and aspiration criterion) on the performance of the TS based heuristic. Using the techniques of evolution curves, and response tables and response graphs, the best combination of the factors for the TS based heuristic algorithm is identified. Empirical results of the comparison of the proposed TS based heuristic algorithm with the best known heuristic to solve the problem are reported and discussed.

Keywords: flowshop scheduling, minimum flow time, tabu search, empirical evaluation.

*Address for correspondence:
Department of Management, Ball State University
Muncie, IN 47306, USA.
e-mail: jgupta@bsu.edu
Fax: 765-285-8024

DESIGNING A TABU SEARCH ALGORITHM TO MINIMIZE TOTAL FLOW TIME IN A FLOW SHOP

1. INTRODUCTION

Consider the following flowshop problem: at time zero, a set $N = \{1, 2, \ldots, n\}$ of n independently available jobs are to be processed by a given set $M = \{1, 2, \ldots, m\}$ of m machines in the same technological order, first on machine 1, then on machine 2, ..., and last on machine m. Preemption of jobs is not allowed. The processing time of job $i \in N$ at machine $j \in M$ is represented by p_{ij} and includes its setup time at machine j. Define a permutation schedule $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$ as the common processing order for each machine $j \in M$ which implies that job $\pi(i)$ is processed on each machine before job $\pi(i+1)$ for all $i \leq (n-1)$. Then, the completion time of job $\pi(i)$ at machine j, $C_j(\pi(i))$, is given by the following recursive relationship:

$$C_j(\pi(i)) = \max\{C_j(\pi(i-1)); C_{j-1}(\pi(i))\} + p_{i,\pi(i)} \tag{1}$$

where $C_j(\pi(0)) = C_0(\pi(i)) = 0$ for all $i \in N$ and $m \in M$.

The total flow time of schedule π, $F(\pi)$ is given by the following expression:

$$F(\pi) = \sum_{i=1}^{n} C_m(\pi(i)). \tag{2}$$

The flowshop scheduling problem considered in this paper is one of finding a schedule π such that the sum of the flow (completion) times, $F(\pi)$, calculated by using Equations (1) and (2) above is minimum. Using the standard three field notation [1, 2], this problem can be represented as a $F||\sum C_i$ problem where F denotes a flowshop with arbitrary number of machines and $\sum C_i$ denotes that the objective function is the minimization of the total flow time of all jobs. Contrary to Johnson's [3] result that the $F2||C_{\max}$ problem involving the minimization of the makespan (*maximum completion time*) of all jobs in a two-machine flowshop can be polynomially solved, the $F||\sum C_i$ problem is NP-hard in the strong sense even for the two-machine case [4]. Owing to its complexity, implicit enumeration and heuristic approaches have been developed to solve the $F||\sum C_i$ problem.

Ignall and Schrage [5] developed a branch and bound algorithm to solve the $F2||\sum C_i$ problem which was extended to solve the $F||\sum C_i$ problem by Bansal [6]. A dominance based combinatorial algorithm was developed by Gupta [7]. Several constructive heuristic algorithms to solve this problem have been developed, notable among them are the two heuristics by Ho and Chang [8] and Rajendran [9]. However, the exact optimization algorithms are not computationally efficient for solving problems involving more than 15 jobs and the effectiveness of heuristics for optimizing the $F2||\sum C_i$ problem is not really known.

In this paper, we discuss the design of a tabu search based heuristic to solve the $F||\sum C_i$ problem. Our emphasis is more on the process of designing a tabu search algorithm to solve the problem rather than to investigate the efficacy of the use of advanced and contemporary features (like memory functions and dynamic list sizes discussed by Glover [10] and Glover and Laguna [11, 12]) of Tabu Search approach.

The rest of the paper is organized as follows. Section 2 describes the tabu search based heuristic and various factors affecting its performance in finding an optimal schedule. Using the evolution curves and response tables and graphs, Section 3 describes the computational results of factorial experiments to select the best combination of various factors to maximize the effectiveness of the TS based heuristic algorithm for solving the $F||\sum C_i$ problem. The effectiveness of the proposed TS based heuristic algorithm in finding an optimal solution to the $F||\sum C_i$ problem is then empirically evaluated in Section 4. Finally, Section 5 summarizes the major findings of this paper and provides some fruitful directions for future research.

2. TABU SEARCH BASED HEURISTIC

Tabu search involves the exploration of a problem's solution space through the iterative investigation of solution neighborhoods. Movement from one neighborhood to the next is prompted by the search for an improved solution guided by the evaluation of an objective function to be optimized. The search is terminated when some predefined stopping criterion has been met.

Compared with traditional iterative search processes, the movement from one solution, s, to another solution, s', in a tabu search does not require that the objective function value of this new solution, $f(s')$, be better than that of the current solution, $f(s)$. Another distinguishing factor of a tabu search is its use of a tabu list to keep track of solutions selected in past iterations. At each iteration, only a selected number of moves is posted to a tabu list containing a finite number of slots. The tabu list is manipulated like a FIFO queue in that each time a solution is posted to one end of the list, it pushes a solution out the other end. This results in a list that can be used to prohibit past solutions from reappearing for a period of time equivalent to the number of iterations corresponding to the size of the list (number of slots). These distinguishing features aim at preventing cycling of the search and expanding the diversity of the resulting search to uncharted regions of the solution space away from local optima.

However, the very nature of the tabu list may constrain the search prohibiting the exploration of solution regions which are appealing. To overcome this possible problem, Glover [13] uses an aspiration criterion to determine if a tabued move offers the potential of leading the search to a more promising region. If so, then that move is removed from the list (aspired) and used. While several aspiration criteria are available to measure the potential of a tabued move, the most commonly used aspiration criterion is to aspire a tabued move if it can provide a better solution than the incumbent solution.

Tabu search was introduced by Glover in the 1980s [13], and has been applied to a wide variety of problems [11, 12]. Its diverse applications include neural networks training, transportation as in vehicle routing and travelling salesman problems, layout as in quadratic assignment problems, graphs as in coloring, partitioning, and clustering problems, probabilistic logic and expert systems, telecommunications as in path assignment and bandwidth packing, circuit design, and computer design. In recent years, scheduling has been recognized as a potential area for the application of tabu search. Tabu search has been applied to different types of scheduling problems ranging from employee to production problems.

For permutation flowshop problems, Widmer and Hertz [14] and Taillard [15] used Tabu Search to minimize makespan. Adenso-Días [16] conducted a study with minimizing weighted tardiness as the objective function and Kim [17] experimented with minimizing mean tardiness as the performance measure. Skorin-Kapov and Vakharia [18] applied TS to the flow-line manufacturing cell problem to minimize makespan. They compared TS with simulated annealing (SA) and showed that TS provided better solutions in less computation time. Reeves [19] showed that the TS method gave better results than the SA approach when applied efficiently in a general machine sequencing problem. A review of the use of Tabu search and other local search algorithms to solve the scheduling problems is provided by Anderson, Glass, and Potts [20]. Gupta, Palanimuthu, and Chen [21] designed a Tabu Search based heuristic for solving a two-machine flowshop problem with a secondary criterion. Nowicki [22] developed a tabu search approach to solve the permutation flowshop scheduling problem with finite buffers.

Based on the discussion above, a tabu search procedure for a minimization problem may be summarized below:

begin Tabu Search
$iter = 1$;
generate initial solution i;
evaluate $f(i)$;

```
define incumbent solution f* = f(i);
initialize tabu list L(iter);
while {not termination} do
begin neighborhood search;
   best_move = ∞;
   repeat
      { if (new_move < best_move) then
         { if (new_move < f* or new_move ∉ L(iter)
            then { best_move = new_move}
               }
            }
   until {specified number of neighbors}
   if (best_move < f*) then
      f* = best_move
   update L(iter);
   iter = iter + 1;
end neighborhood search;          end Tabu Search
```

In the above description of the tabu search approach, $f(i)$ represents the resulting objective value of solution (move) i, f^* represents the objective function value of the incumbent solution for the neighborhood search, and $L(iter)$ is the current list of tabued moves at the given iteration.

We now discuss various factors of tabu search that significantly contribute to its performance in finding an optimal solution to the $F||\sum C_i$ problem. Based on past research [14, 15], the performance of a tabu search is highly dependent on the following factors: initial solution, type of move, neighborhood size, tabu list size, stopping criterion, and aspiration criterion. We briefly describe each factor in relation to its effect in solving the $F||\sum C_i$ problem.

2.1. Initial Solution

This is a point in the solution space from where the search process begins. The initial solution of tabu search may be generated randomly (referred to as RAN) or by using some known polynomially bounded heuristic. In this research, the effect of these two methods of generating a initial solution on the efficiency of the TS heuristic was investigated. And for the heuristic initial solution, Rajendran's heuristic [9], marked by RAJ, was used.

2.2. Type of Move

A move is an event that transforms the search from current solution to its neighboring solution. For production scheduling problems, there are two common types of moves: insert, a move which removes a job from the jth position and inserts it at the ith position, and swap, a move which interchanges the jobs in the ith and jth position, commonly known as the pair-wise interchange method. In this study, the results of the TS based heuristic was investigated using these two types of move, denoted as INSERT and SWAP, respectively.

2.3. Neighborhood Size

This represents the number of candidate solutions to be evaluated at each iteration of the search process. Taillard [15] describes three different ways of evaluating a neighborhood. The first kind is to examine all the possible neighboring solutions and select the best move that is not tabu as the candidate for the next search.

This complete examination of neighborhood may lead to high quality solutions but the computational time required may be high. The second kind is to randomly survey the neighbors and select the first improving, non-tabu solution as the candidate solution for the next search. However, if there is no improving solutions, then the whole neighborhood has to be explored. The third kind is to randomly examine a fraction of the neighborhood, and select the best solution as a candidate for next search. While Reeves [19] found that for most scheduling problems, the third approach described above provide best results, Taillard [15] points out that this third method of examination may not be suitable for solving the flowshop problems. Therefore, the first two types of neighborhood examinations described above, denoted as WHLSIZE and RANSIZE, respectively, were used to analyze the efficiency of the TS based heuristic algorithm.

2.4. Tabu List Size

The key element of tabu search is the tabu list. It contains the list of moves executed and therefore tabued. Too small a tabu list may cause cycling of the search, while too large a list may prohibit TS to certain good solution regions. Past applications [13, 21] showed that, independent of problem size and structure, tabu list size lies in the range of 5 to 12, with 7 as the most common occurrence. In this study, tabu list sizes of 5, 7, and 12, respectively denoted as LS5, LS7, and LS12, were investigated.

2.5. Aspiration Criterion

An aspiration criterion is used to override the tabu status of a move so as to prevent rejection of a good solution due solely to its tabu status. In this experimentation, aspiration level is not varied. The aspiration level used is as follows: override the tabu status of a move if it yields a better solution than the incumbent solution.

2.6. Stopping Criterion

In order to terminate the search process, two of the most common and simplest stopping conditions used are:

- Stop if the number of iterations is greater than a constant value, NITER, and
- Stop if the number of iterations without improving the incumbent solution is greater than a constant value, NOIMP.

In our preliminary experiments, we found that the use of fixed number of iterations at NITER= 200 established the convergence of search. Therefore, in this study, we used the first terminating criterion with NITER = 200. Moreover, the second criterion may be efficient in speed, but since the number of iterations with no improvement will be affected by the complexity of the solution space and the problem size, a suitable number of iterations cannot be determined in advance of the application of the TS based heuristic algorithm.

3. COMPUTATIONAL RESULTS

In order to completely analyze the effects of these factors on the performance of a tabu search and to identify the best combination of these factors for a tabu search approach to solve the $F||\sum C_i$ problem, a full factorial experiment was designed. Since the stopping condition was set to 200 iterations and a simple form of aspiration criterion was used, a complete experiment of the remaining factors encompasses $24 = (2^3 \times 3^1)$ combinations of the factors: two approaches of generating initial solutions (RAN and RAJ), two types of move (INSERT and SWAP), two neighborhood sizes examined in each iteration (WHLSIZE and RANSIZE), and three list sizes (LS5, LS7, and LS12). Through this design, we can analyze the effect of each factor on the performance of tabu search, and identify a good combination of the factors that will enhance the efficiency of the application of tabu search.

The performance of the TS based heuristic was evaluated for nine groups of problems. Each problem group is a combination of the number of jobs (10, 20, and 30) and the number of machines (5, 15, and 25). Ten different problems were solved for each of the nine groups, with the processing times for each problem generated randomly

and distributed uniformly between 1 and 99. Each problem, in turn, was solved using the TS based heuristic for the 24 different combinations discussed earlier. The program was coded in FORTRAN 77, and executed on SUN4/490 machine.

The problem groups with 25-machines (*i.e.* 10-job, 25-machine, 20-job, 25-machine, and 30-job, 25-machine) were considered to analyze the factor settings for the TS based heuristic. This is because a large-size problem will provide a larger and more complex solution space, and through this, adequate information regarding the application of tabu search to the $F||\sum C_i$ problem can be obtained. The performance of the TS based heuristic with the best factor setting will then be evaluated by comparing its solutions with that of Rajendran's heuristic.

Generally, in conventional design of experiments, analysis of variance is used to find the effects of the factors and to determine the optimum combination of the factors. However, analysis of variance assumes: (1) normality of output distribution for each combination of the factors and (2) equality of variance of each distribution to one another. Since the output of some scheduling problem may not satisfy the above assumptions, alternate approaches to analyze the effect of various experimental factors were explored. Tukey analysis suggested by Games and Howell [23] and Klockars and Sax [24] could have been used for this purpose. However, we decided to use the evolution curve analysis, and response graph analysis to analyze the output for the $F||\sum C_i$ problem as this method provides a better graphical view of the process and results. Further, this method can be used even if the standard assumptions for the analysis of variance are satisfied. For this reason, we did not test if the previously stated two assumptions were satisfied.

3.1. Evolution Curve Analysis

An evolution curve depicts the evolution of the TS based heuristic for the factors under consideration. There were 24 evolution curves (corresponding to the 24 different combinations of the factors) plotted for each of the 3 groups of problems. Each curve represents the average total flow time of the 10 different problems at each iteration of the TS based heuristic. The following paragraphs briefly discuss the evolution curves for the group of 30-job problems used in this study.

To proceed with the analysis, the 24 curves were classified into figures according to each of the four factors. For instance, when the factor considered was the initial solution, twelve figures were plotted, each displaying the two curves RAN and RAJ (two levels of the factor of initial solution) under the 12 different combinations of the levels of the other three factors (type of move, size of neighborhood evaluated, and tabu list size). Similarly, twelve figures were plotted based on each of the factors: type of move and neighborhood size, and eight figures were plotted based on tabu list size. Examining these four groups of figures, we found that the results of the figures based on the initial solution and type of move were consistent, but not for neighborhood size and tabu list size. All the twelve figures based on initial solution showed that RAJ dominated RAN, and all the twelve figures based on type of move showed that INSERT dominated SWAP. Therefore, the best levels of the factors, initial solution and type of move, were set at RAJ and INSERT, respectively.

Figure 1 displays the curves of RAN and RAJ under SWAP (type of move), WHLSIZE (neighborhood size), and list size LS5. Figure 2 displays the curves of INSERT and SWAP under RAJ, RANSIZE, and LS12.

Further analysis was conducted based on the condition that RAJ and INSERT were chosen. The groups of figures corresponding to neighborhood size and tabu list size were examined. All these figures showed that RANSIZE dominated WHLSIZE, and LS12 dominated LS5 and LS7. Therefore, the best levels for these two factors were set at RANSIZE and LS12, respectively. Figure 3 exhibits the curves of WHLSIZE and RANSIZE under RAJ, INSERT, and LS12, and Figure 4 exhibits the curves of tabu list size under RAJ, INSERT, and RANSIZE. These figures show that WHLSIZE converges faster than RANSIZE, but RANSIZE eventually converges to a better solution than WHLSIZE. This implies that WHLSIZE can provide more information about the neighborhood of a solution in each iteration, so a search using WHLSIZE can lead to better solutions faster;

however, since WHLSIZE may lack diversity in searching the solution space, the search may be limited to some small solution region and be trapped in local optimality.

A similar analysis was performed on the 10-job and 20-job problem groups. The conclusion for the 20-job problem group is the same as that for the 30-job problem group; however, the results of the 10-job problem group are different. None of the levels of the factors dominate its corresponding level(s) in all the combinations. A further examination was conducted on each problem in this group. It was observed that, in each problem, most of the solutions converged to one or two values, and, the average difference between the best and worst solutions is less than 0.7%. Moreover, the average percentage of identical solutions with 24 different factor combinations for each problem was found to be 65%. Therefore, we may conclude that the TS based heuristic performed very well for this problem group regardless of the factor combinations.

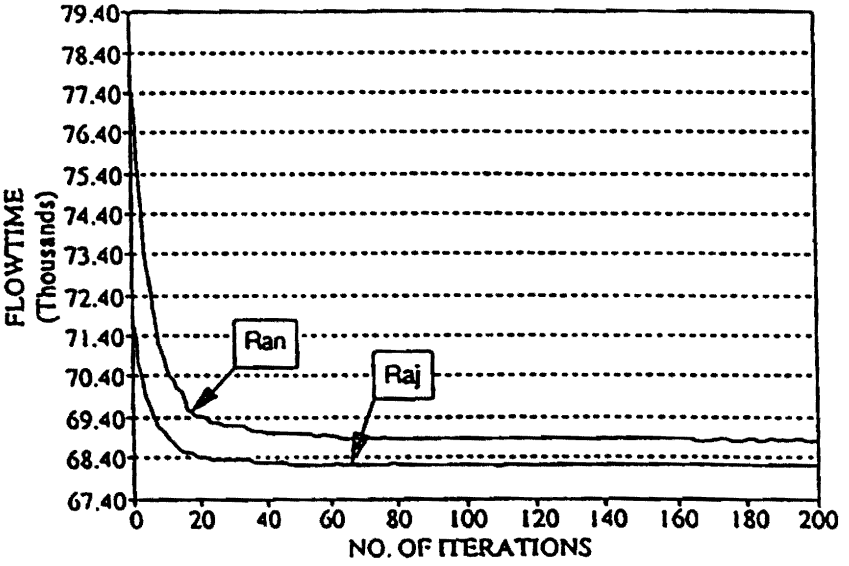

Figure 1. Graph of RAJ and RAN under SWAP, WHLSIZE, and LS5.

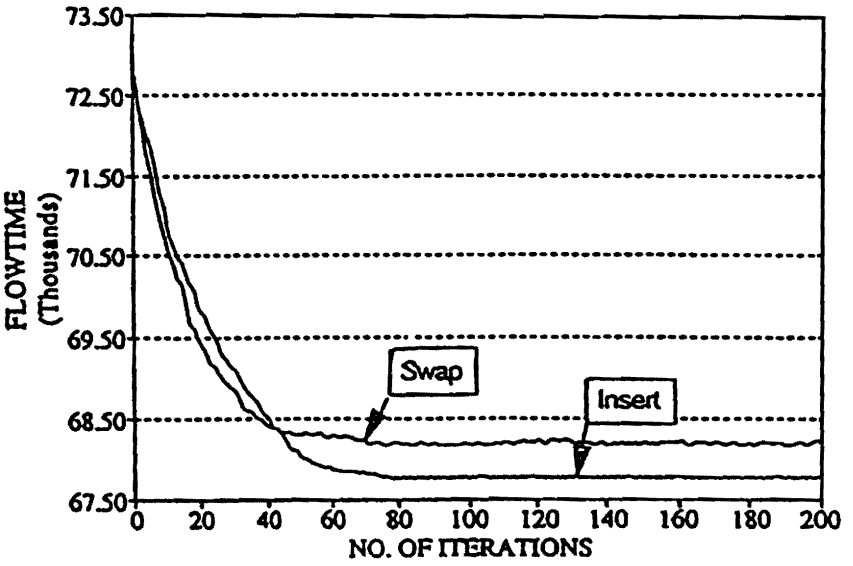

Figure 2. Graph of INSERT and SWAP under RAJ, RANSIZE, and LS12.

From the foregoing analysis, the best combination of the factors for the flow shop problem with total flow time as the criterion is as follows: RAJ (approach for generating initial solution), INSERT (type of move), RANSIZE (size of neighborhood evaluated in every iteration), and tabu list size LS12. This best combination is different from the one by Widmer and Hertz [14] for the $F||C_{\max}$ problem. They found that SWAP for type of move and WHLSIZE for neighborhood size for the tabu search approach provided best results for the solution of the $F||C_{\max}$ problem. Therefore, the best combination of various factors in a TS based heuristic algorithm is problem specific, and needs to be justified before we apply tabu search to different problems. Further, this analysis shows that more research in developing polynomially bounded heuristic algorithms is valuable as it may provide good initial solutions, and assist tabu search to find a good final solution, especially when the problem size is large.

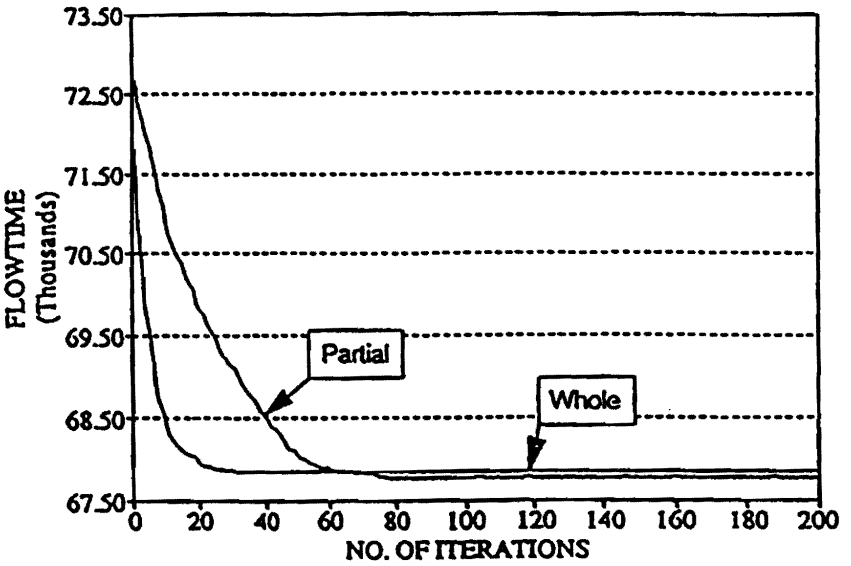

Figure 3. Graph under WHLSIZE and RANSIZE under RAJ, INSERT, and LS12.

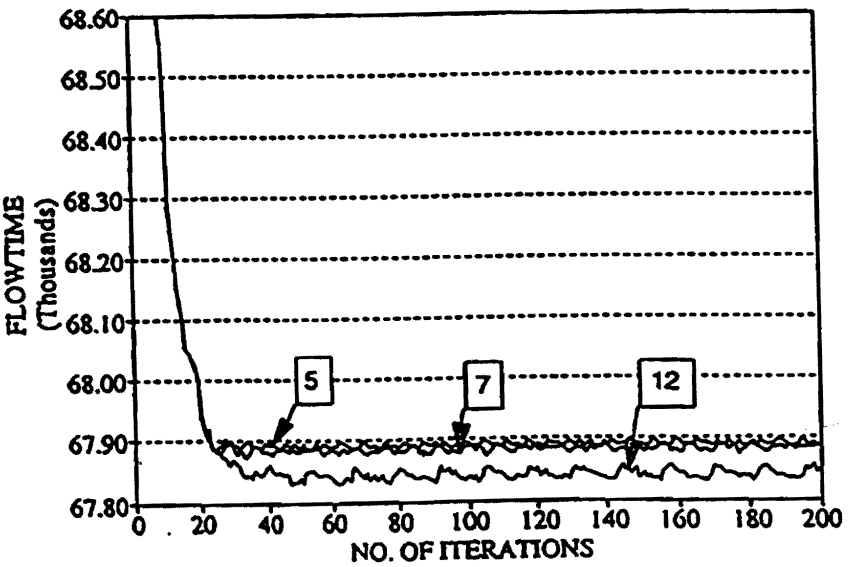

Figure 4. Graph of tabu list size under RAJ, INSERT, and RANSIZE.

3.2. Response Table and Response Graph Analysis

Response table and response graph analysis is a practical analytical technique for factorial experiments as such analysis is often insensitive to the assumptions for analysis of variance. In general, response tables are used to analyze the effects of single factors, and response graphs are used to analyze the effects of two-factor interactions. To develop a response table, the average response (which is the total flow time for the $F||\sum C_i$ problem) for each level of each factor is computed. Then the maximum difference (note that this is an absolute value) between (or among) the average responses of the levels of each factor is calculated. The relative significance of each factor is represented as a percentage of the difference of each factor to the total difference. The higher the percentage of a factor, the stronger the factor. Note that response table and response graph analysis was not conducted for the 10-job problem group because, as described earlier, the TS based heuristic performed very well for the 10-job problems regardless of the factor combinations.

Tables 1 and 2 present the response tables for the 20-job and 30-job problem groups, respectively. The results in both the tables are consistent. Initial solution, type of move, and neighborhood size are strong factors, and tabu list size is a weak factor. RAJ, INSERT, and RANSIZE dominate RAN, SWAP, and WHLSIZE. Furthermore, the effects of initial solution and neighborhood increase when the problem size increases. This

Table 1. Response Table for 20-Job Problem Group.

Factors	Levels	AVR TFT	Difference[a]	Percent[b]
Initial solution	RAN	39166.44	147.86	24.63
	RAJ	39018.58*		
Type of move	INSERT	38939.27*	306.48	51.05
	SWAP	39245.75		
Neighborhood	WHLSIZE	39146.13	107.24	17.86
	RANSIZE	39038.89*		
List size	LS 5	39110.66	38.81	6.46
	LS 7	39095.01		
	LS 12	39071.85*		

*: The better solution.
[a]: The difference between the minimum and maximum solution of the levels in a factor.
[b]: The percentage contribution of the factor to the observed total difference.

Table 2. Response Table for 30-Job Problem Group.

Factors	Levels	AVR TFT	Difference[a]	Percent[b]
Initial solution	RAN	68376.03	381.90	35.03
	RAJ	67994.13*		
Type of move	INSERT	67997.70*	374.90	34.37
	SWAP	68372.40		
Neighborhood	WHLSIZE	68338.53	306.90	28.15
	RANSIZE	68031.63*		
List size	LS 5	68180.83	26.63	2.45
	LS 7	68200.53		
	LS 12	68173.90*		

Footnotes *, [a], [b]: As in Table 1.

confirms the conclusions in the evolution curve analysis: (1) a good initial solution is important for tabu search when the problem size increases; and (2) RANSIZE may improve the capability of diversification of tabu search, and assist tabu search to find a better solution, in particular, when problem size is large.

As mentioned, response graphs are also used to analyze the effects of 2-factor interactions. There are six 2-factor interactions for the 4-factor design (the number of ways to select 2 factors out of 4 factors). To plot the response graph for a 2-factor interaction, the vertical axis represents the response of the experiment; one of the two factors under consideration serves as the label for the horizontal axis and the other factor is presented with a number of line segments (according to its number of levels of the factor). Each vertex of the line segments represent the average response under each combination of the factors. For instance, Figure 5 is the response graph which displays the interaction between the factors: initial solution and type of move for 30-job problems. In this figure, the horizontal axis depicts the two approaches for generating an initial solution and the two line segments represent the two types of move. The vertices of the line segments represent the average of the total flow times of the problems solved under the combinations of the levels of the two factors.

Use of response graphs to study interactions between various factors is based on the relationships between (or among) the line segments in the graphs. If the line segments in a graph are parallel or close to parallel to each other, we will conclude that there is no interaction between the factors considered in the graph; otherwise, there is an interaction between the factors. If a factor has no interaction with other factors in the experiment, the best level of the factor is the level of the factor that provided the better results (this can be easily observed in a response table). However, if a factor has an interaction with another factor, then the best levels of the factors is the combination of the factors that provided the best result (this cannot be obtained in response table, but in response graph).

Figures 5 through 10 present the response graphs for the 30-job problem group. Figures 6 through 9 show that every factor has an interaction with at least one of the other factors. Since the respective factors in Figure 5 and Figure 10 have no interaction, the best combinations for the experiment will be determined with the response graphs in the remaining four figures 6 through 9. The dominant combinations in these four figures 6 through 9 are RAJ and RANSIZE, RAJ and LS5, INSERT and RANSIZE, and INSERT and LS12, respectively. It is

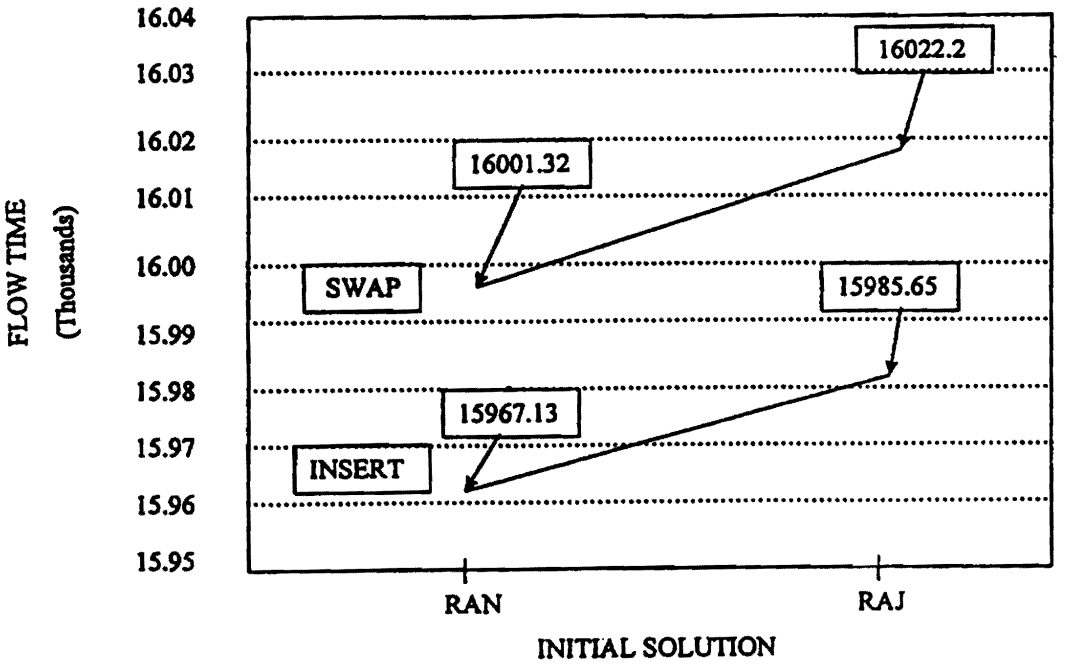

Figure 5. Interaction between initial solution and type of move.

clear that RAJ, INSERT, and RANSIZE will be assigned to the best combinations for the experiment, and tabu list size needs to be further studied. Figure 7 shows that LS5 dominates LS12 by 26.68 (68000.88 - 67974.2) under RAJ, but Figure 9 shows that LS12 dominates LS5 by 90.83 (68035.33 - 67944.7) under INSERT. Thus, we may expect that the output under RAJ, INSERT, RANSIZE, and LS12 will be superior to the output under

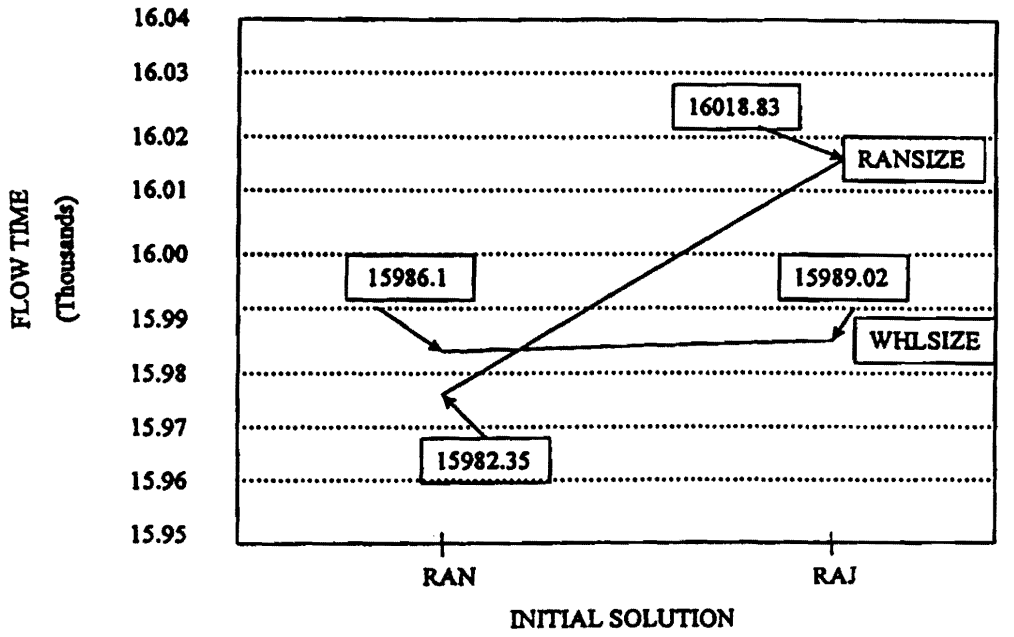

Figure 6. Interaction between initial solution and size of neighborhood evaluated.

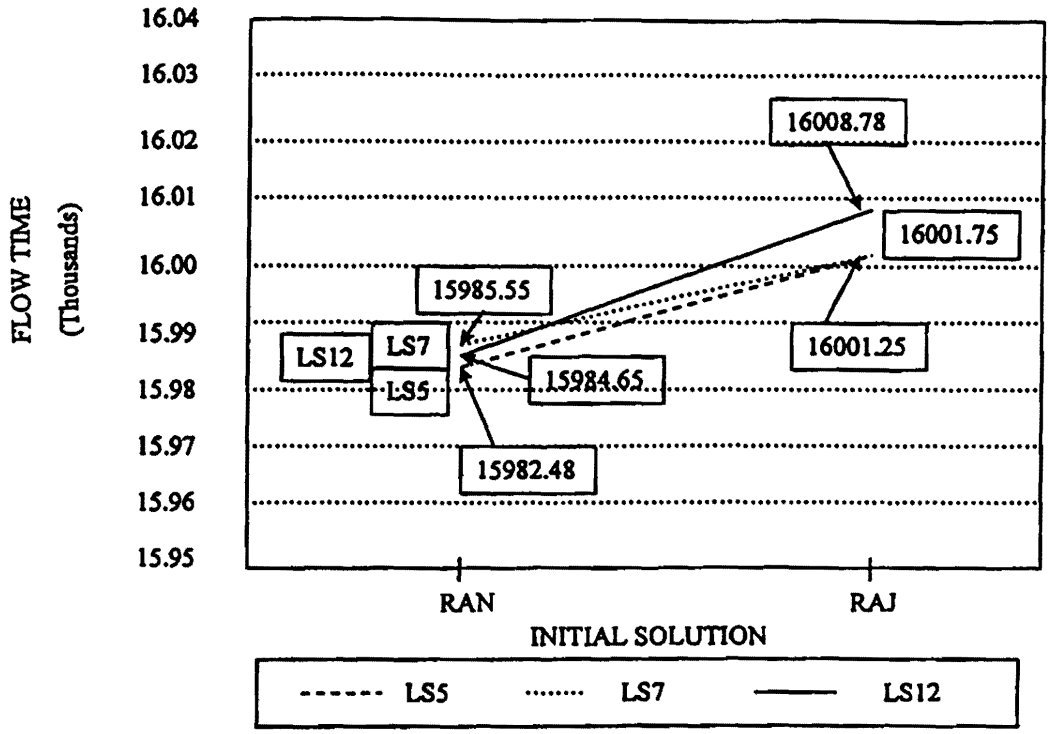

Figure 7. Interaction between initial solution and size of neighborhood evaluated.

RAJ, INSERT, RANSIZE, and LS5. Hence, RAJ, INSERT, RANSIZE, and LS12 are the best combination for 30-job problems. The same analysis was conducted for the 20-job problem group, and the same conclusion was obtained.

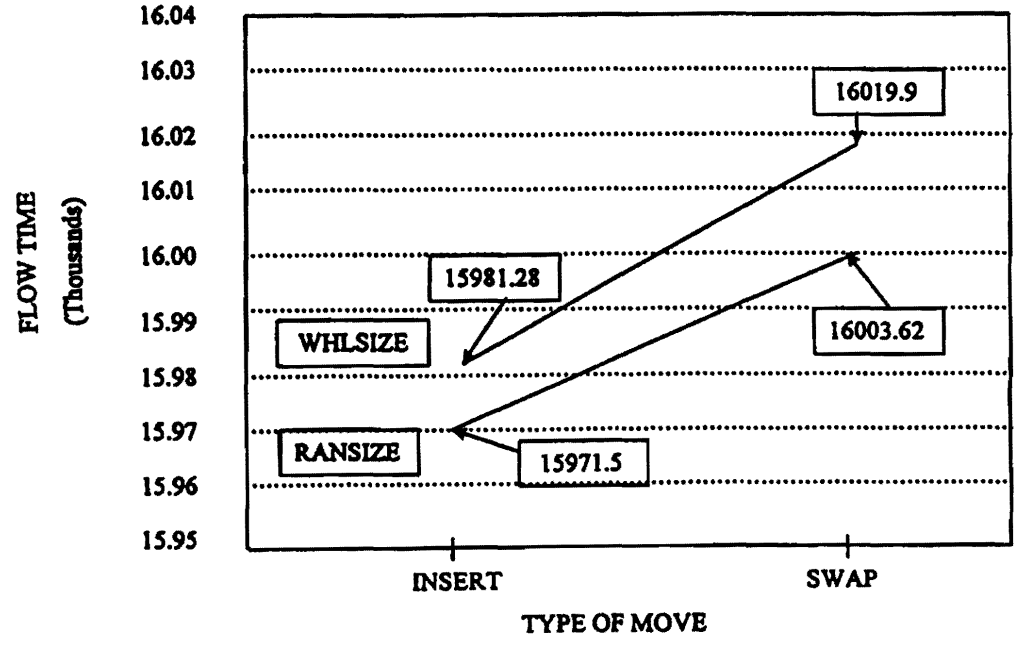

Figure 8. Interaction between type of move and size of neighborhood evaluated.

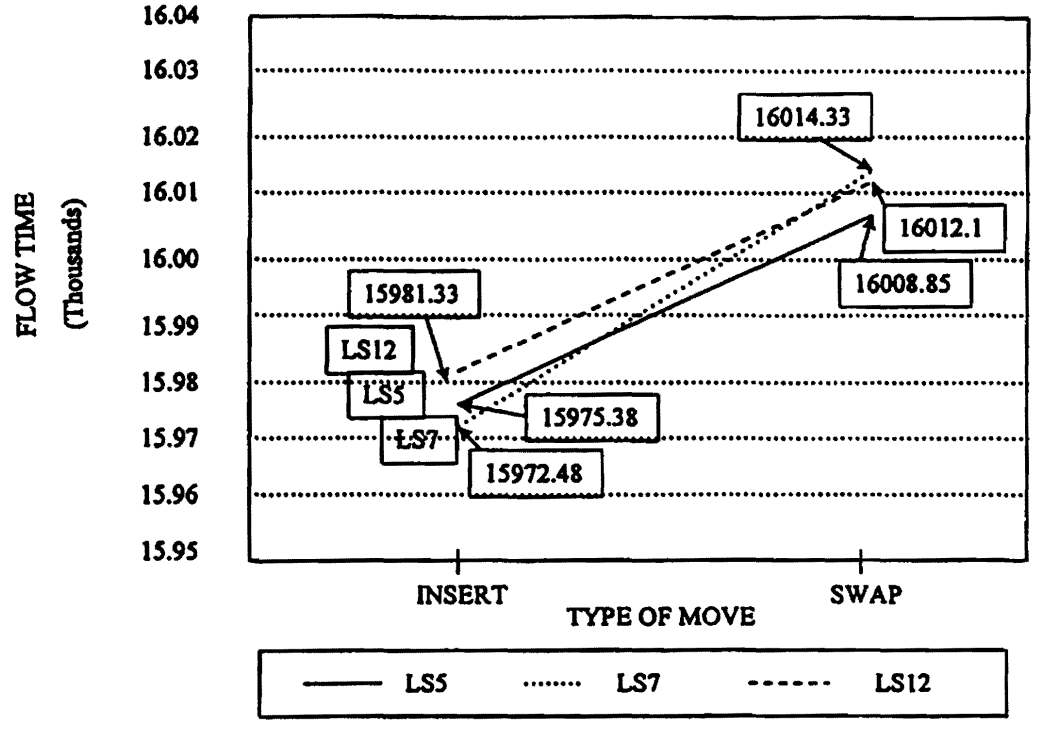

Figure 9. Interaction between type of move and tabu list size.

It is noted that the best combination of various factors resulted from response graph analysis are the same as that from an evolution curve analysis for the $F||\sum C_i$ problem. In addition, response graph analysis also provides some insight into the performance of tabu search. For instance, for the present problem, Figure 6 shows that the effect of neighborhood size under RAN is much stronger than that under RAJ. Thus, in the application of TS based heuristic using RAJ for generating the initial solution, neighborhood size is not likely to strongly affect performance of the TS based heuristic. However, the use of RAN for generating the initial solution may lead to an opposite conclusion. In addition, Figure 9 shows that LS12 dominates LS5 under INSERT, but LS5 dominates LS12 under SWAP. Thus, with the use of INSERT as the type of move in applying the TS based heuristic, LS12 is better than LS5; however, use of SWAP may lead to the opposite conclusion. This illustrates the need to consider interactions between various factors; otherwise, the conclusions may be misleading and the performance of tabu search may be negatively influenced.

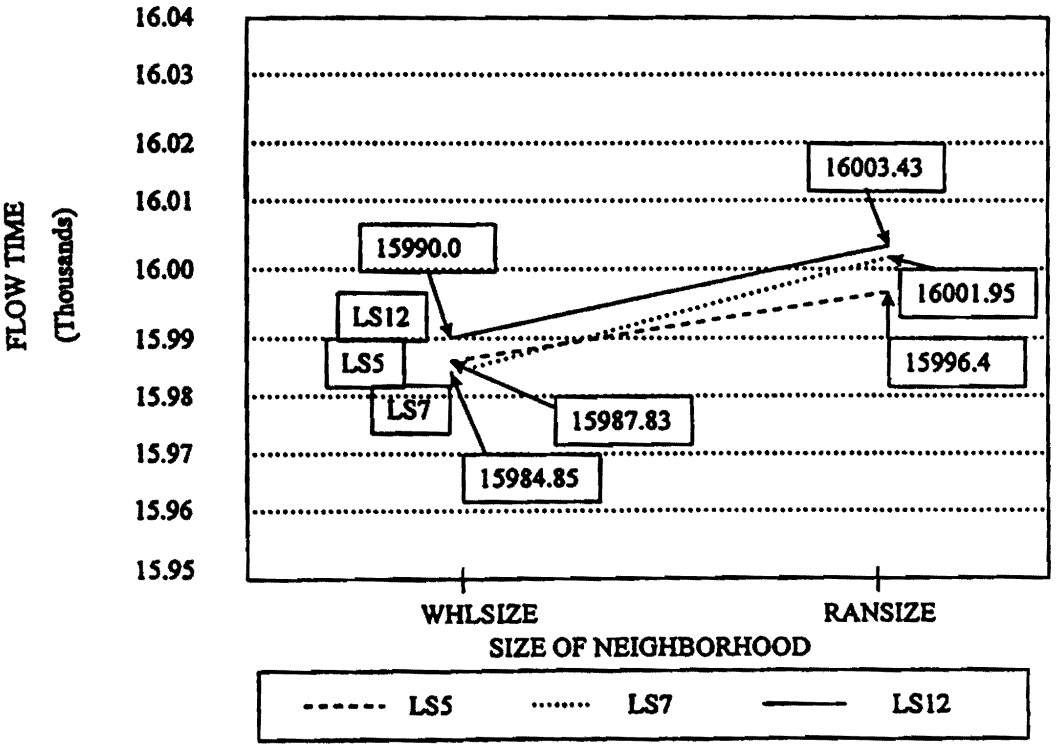

Figure 10. Interaction between size of neighborhood evaluated and tabu list size.

4. EVALUATION OF THE TS BASED HEURISTIC

Using the best combination of various factors found by the analysis of the evolution curves, response tables and response graphs, the solutions obtained from the TS based heuristic are compared with those from Rajendran's heuristic, the best existing heuristic for the $F||C_{\max}$ problem. To make this comparison, 10 new problems were generated (using the same method mentioned earlier) for each of the following 15 groups of problems: 15 combinations of the number of jobs (10, 20, 30, 40, and 50) and the number of machines (5, 15, and 25). Further, to eliminate the bias that Rajendran's heuristic is used as the initial solution, both RAJ and RAN were used as initial solutions.

Table 3 lists the average solution for Rajendran's heuristic, and the two TS based heuristics (called TS-RAN when RAN is used as the initial solution and TS-RAJ when RAJ is used to generate an initial solution) and the average percentage improvements over Rajendran's solution for the two TS-based heuristics, α_{RAJ} and α_{RAN}. The results in Table 3 clearly show that the TS based heuristic with the best factor setting gives significant

improvement over Rajendran's heuristic even when random sequence is used as an initial solution. Moreover, the improvement increases when problem size increases. Thus, even when there is no initial information about the problem, a tabu search based heuristic algorithm provides significantly better solutions than Rajendran's heuristic. This implies that tabu search is a good technique for the $F||\sum C_i$ problem even when there is no initial information about the problem.

Table 3. Average Percentage Improvement of the TS Heuristic over Rajendran's Heuristic.

Job size $n \times m$	Rajendran's solution	TS-RAJ solution	TS-RAN solution	α_{RAJ} Value	α_{RAN} Value
10 × 5	4900.5	4765.6	4738.4	2.83	3.42
10 × 15	10760.1	10383.4	10386.0	3.63	3.60
10 × 25	16520.9	15990.5	15950.8	3.32	3.57
20 × 5	14567.6	13450.7	13670.8	8.30	6.56
20 × 15	29593.7	26978.4	27174.8	9.69	8.90
20 × 25	41644.0	38756.7	38866.3	7.45	7.16
30 × 5	30290.4	27714.1	28202.4	9.30	7.40
30 × 15	53737.8	49183.0	50139.8	9.26	7.18
30 × 25	73269.2	67741.8	67912.2	8.15	7.89
40 × 5	50186.9	45027.0	45238.9	11.46	10.94
40 × 15	84520.3	76116.7	75749.2	11.04	11.58
40 × 25	113578.4	103817.1	102859.0	9.40	10.42
50 × 5	75770.5	67403.9	68162.5	12.41	11.16
50 × 15	122477.9	108350.5	109243.6	13.04	12.11
50 × 25	157380.7	142620.9	143021.4	10.35	10.04

The computational time required to solve the problems was not measured. For the $F||\sum C_i$ problem with n jobs and m machines, the complexity of the Rajendran's heuristic used in this study is $O(n^3m)$. For the same problem, the complexity of the proposed TS based heuristic algorithm using RAN as the initial solution is $O(Kn^2m)$ where K is a constant that depends on the list size and the number of iterations. Since the list size and the number of iterations did not vary with the size of the problem, the computational complexity of the proposed TS based heuristic algorithm is less than that of Rajendran's heuristic. Therefore, on the average, the CPU time required to solve large-sized problems by the proposed algorithm will be less than that required by Rajendran's heuristic.

The results in Table 3 also indicate that Rajendran's heuristic algorithm may well not be very effective in finding optimal solutions to the $F||\sum C_i$ problem as a TS based heuristic provides a significant improvement in solution quality. Considering that a very simple Tabu Search procedure was used, it is therefore, clear that the use of the advanced and contemporary features (like memory functions and dynamic list sizes discussed by Glover [10] and Glover and Laguna [11, 12]) of Tabu Search approach will enhance the effectiveness and efficiency of the TS based heuristic algorithm to solve the $F||\sum C_i$ problem.

5. CONCLUSIONS

This paper has discussed a process for designing a TS based heuristic for solving the flow shop problem with total flow time as the criterion. We designed a factorial experiment to completely analyze the effects of four different factors (initial solution, type of move, size of neighborhood, and list size) on the performance of the TS based heuristic for solving the $F||\sum C_i$ problem. The results of the experiment were studied using evolution curve analysis, and response table and response graph analysis. These analyses illustrated the relative contribution of each factor on the performance of the TS based heuristic and identified the best combination of the factors for the TS based heuristic for the candidate problem.

The effectiveness of the TS based heuristic with the best combination of the factors was evaluated by comparing the solutions of the TS based heuristic with those of Rajendran's heuristic for the $F||\sum C_i$ problem. This comparison showed that the TS based heuristic significantly improves the solutions of Rajendran's heuristic even when the initial solution was generated randomly. Thus, the proposed TS based heuristic algorithm is quite effective in finding an optimal solution to the $F||\sum C_i$ problem and indicates that Rajendran's heuristic may not be very effective in minimizing total flow time in a flowshop problem. Since no advanced and contemporary features (like memory functions and dynamic list sizes) of Tabu Search approach were used in this research, it is clear that the augmentation of such modern developments in Tabu Search will further improve the quality of the solution obtained by the TS based heuristic algorithm.

Two other important findings in this research are as follows: first, the best combination of the factors for the application of TS is problem specific and may significantly affect the performance of TS based heuristic algorithm to solve a problem. Therefore, to apply TS to a certain class of problems, the factor combination have to be justified in advance. Second, the research in developing polynomially bounded heuristic algorithms is valuable for it may provide good initial solution for TS and yield a better final solution.

This research also indicates some directions for future research. Firstly, extension of the proposed TS based heuristic to solve m-stage flowshop problems with a secondary criterion is both interesting and useful. Secondly, finding the best combination of various factors for a TS based heuristic for several other criteria will extend the utility of TS for solving scheduling problems. Finally, use of a TS based heuristic should be explored for finding an improvement potential of various polynomially bounded scheduling heuristics.

References

[1] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, "Sequencing and Scheduling: Algorithms and Complexity", in *Logistics of Production and Inventory.* ed. S.C. Graves, A.H.G. Rinnooy Kan, and P.H. Zipkin. Amsterdam: North Holland, 1993, pp. 445–522.

[2] B. Chen, C.N. Potts, and G.J. Woeginger, "A Review of Machine Scheduling: Complexity, Algorithms and Applications", in *Handbook of Combinatorial Optimization.* ed. D-Z Du and P. M. Pardalos. Dordrecht: Kluwer, 1998, pp. 21–169.

[3] S.M. Johnson, "Optimal Two and Three Stage Production Schedules with Setup Times Included", *Naval Research Logistics Quarterly*, **1** (1954), pp. 61–68.

[4] M.R. Garey, D.S. Johnson, and R. Sethi, "The Complexity of Flowshop and Jobshop Scheduling", *Mathematics of Operations Research*, **1** (1976), pp. 117–129.

[5] E. Ignall and L.E. Schrage, "Application of Branch and Bound Technique to Some Flow-shop Problem", *Operations Research*, **13** (1965), pp. 400–412.

[6] S.P. Bansal, "Minimizing the Sum of Completion Times of n-Jobs Over m-Machines in a Flowshop", *AIIE Transactions*, **9** (1977), pp. 306–311.

[7] J.N.D. Gupta, "Optimal Scheduling in a Multi-stage Flowshop", *AIIE Transactions*, **4** (1972), pp. 238–243.

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

[9] C. Rajendran, "Heuristic Algorithm for Scheduling in a Flowshop to Minimize Total Flow Time", *International Journal of Production Economics*, **29** (1993), pp. 65–73.

[10] F. Glover, "Tabu Search and Adaptive Memory Programming-Advances, Applications and Challenges", in *Interfaces in Computer Science and Operations Research.* ed. R.S. Barr, R.V. Helgason, and J.L. Kennington. Dordrecht: Kluwer, 1986, pp. 1–75.

[11] F. Glover and M. Laguna, "Tabu Search", in *Modern Heuristic Techniques for Combinatorial Problems.* ed. C. Reeves. Oxford: Blackwell, 1993, pp. 70–150.

[12] F. Glover and M. Laguna, *Tabu Search.* Dordrecht: Kluwer, 1997.

[13] F. Glover, "Tabu Search, part I", *ORSA Journal of Computing*, **1** (1989), pp. 190–206.

[14] M. Widmer and A. Hertz, "A New Heuristic Method for the Flowshop Sequencing Problem", *European Journal of Operational Research*, **41** (1989), pp. 186–193.

[15] E. Taillard, "Some Efficient Heuristic Methods for the Flowshop Sequencing Problem", *European Journal of Operational Research*, **47** (1990), pp. 65–74.

[16] B. Adenso-Días, "Restricted Neighborhood in the Tabu Search for the Flowshop Problem", *European Journal of Operational Research*, **62** (1992), pp. 27–37.

[17] Y. Kim, "Heuristics for Flowshop Scheduling Problems Minimizing Mean Tardiness", *Journal of Operational Research Society*, **44** (1993), pp. 19–28.

[18] J. Skorin-Kapov and A.J. Vakharia, "Scheduling a Flow-line Manufacturing Cell: a Tabu Search Approach", *International Journal of Production Research*, **31** (1993), pp. 1721–1734.

[19] C.R. Reeves, "Improving the Efficiency of Tabu Search for Machine Sequencing Problems", *Journal of Operational Research Society*, **44** (1993), pp. 375–382.

[20] E.J. Anderson, C.A. Glass, and C.N. Potts, "Machine Scheduling", in *Local Search in Combinatorial Optimization.* ed. E. Aarts and J.K. Lenstra. Chichester, England: Wiley, 1997, pp. 361–414.

[21] J.N.D. Gupta, N. Palanimuthu, and C.-L. Chen, "Designing a Tabu Search Algorithm for the Two-Stage Flow Shop Problem with Secondary Criterion", *Production Planning and Control: An International Journal*, **10** (1999), pp. 251–265.

[22] E. Nowicki, "The Permutation Flow Shop with Buffers: A Tabu Search Approach", *European Journal of Operational Research*, **116** (1999), pp. 205–219.

[23] P.A. Games and J.F. Howell, "Pairwise Multiple Comparison Procedures with Unequal N's and/or Variances", *Journal of Educational Statistics*, **1** (1976), pp. 113–125.

[24] A.J. Klockars and G. Sax, *Multiple Comparisons.* San Francisco: Sage, 1986.

Paper Received 10 November 1997; Revised 30 August 2000; Accepted 11 October 2000.