FBE-IE-

A TABU SEARCH ALGORITHM FOR

PARALLEL MACHINE TOTAL TARDINESS PROBLEM

ÜMIT BILGE*
FURKAN KIRAÇ
MÜJDE KURTULAN
PELIN PEKGÜN

A TABU SEARCH ALGORITHM FOR PARALLEL MACHINE TOTAL TARDINESS PROBLEM

Ümit Bilge*, Furkan Kiraç, Müjde Kurtulan, Pelin Pekgün

Department of Industrial Engineering, Bogaziçi University, Bebek, 80815 Istanbul, Turkey

ABSTRACT

In this study, a Tabu Search (TS) approach to the parallel machine scheduling problem is presented. The problem considered consists of a set of independent jobs to be scheduled on a number of parallel processors to minimize total tardiness. Several surveys on parallel machine scheduling with due date related objectives [1, 2, 3] reveal that the NP-hard nature of the problem renders it a challenging area for many researchers who studied various versions. However, most of these studies have the assumption that jobs are available at the beginning of the scheduling period, which is an important deviation form reality. Here, as well as distinct due dates and ready times, features such as sequence dependent setup times and different processing rates for machines are incorporated into the classical model. These enhancements approach the model to the actual practice at the expense of complicating the problem further. The motivation of this study has been to explore the ability of Tabu Search to overcome these difficulties superimposed on the traditional parallel machine scheduling problem.

In order to obtain a robust search mechanism, several key components of TS such as candidate list strategies, tabu classifications, tabu tenure and intensification/diversification strategies are investigated. Alternative approaches to each of these issues are developed and extensively tested on a set of problems obtained from the literature. Considerably better results are obtained and the success of the totally deterministic TS algorithm implemented is thereby demonstrated.

Keywords: Parallel machine scheduling; Total Tardiness minimization; Sequence dependent setup times; Tabu search.

1 INTRODUCTION

1.1 Problem Definition

The classical parallel machine total tardiness problem can be stated as follows: There are n jobs to be processed on m continuously available identical parallel machines. Each machine can process only one job at a time, and each job can only be processed on only one machine. Each job is ready at the beginning of the scheduling horizon and has a distinct processing time and a distinct due date. The objective is to determine a schedule such that total tardiness is minimized, where tardiness of a job is the amount of time its completion time exceeds its due date. The problem is NP-hard, even for a single machine, i.e. m=1 (Du and Leung [4]) and exact methods in which the dimensionality problem is acute are mostly limited to special cases like common due dates and equal processing times (i.e. Root [5], Lawler [6], Elmaghraby and Park [7], Dessouky [8]). A large class of heuristics are based on list scheduling where the jobs are first prioritised according to some rule and then dispatched in this order to the machine with the earliest finish time. Such heuristics are proposed by Wilkerson and Irwin [9], Dogramaci and Surkis [10], Ho and Chang [11] and Koulamas [3]. Koulamas [12] also developed a decomposition heuristic and a hybrid simulated annealing heuristic, while Bean [13] applied a genetic algorithm heuristic to the parallel machine total tardiness problem.

In all the studies cited above it is assumed that machines are identical, all jobs are available at time zero and setup times are non-existent. However, in many real-world situations there exist (i) distinct job ready dates, (ii) uniform parallel machines that are capable of processing these jobs at different speeds (i.e. new machines versus old machines) and (iii) sequence dependent setups. In this paper, these features are also incorporated into the model so as to define a problem closer to reality albeit far more complex than the classical one.

When jobs are allowed to have distinct arrival times as well as due dates, different processing rates on machines and sequence dependent setup times, the literature becomes really sparse. There are only two studies reported on this more general problem to our knowledge and both of them deal with minimizing the total earliness-tardiness costs: Serifoglu and Ulusoy [14] present a genetic algorithm while Balakrishnan et al. [15] report a compact mathematical model to solve small sized (up to 10 jobs) problems.

This paper presents a Tabu Search approach to this generalized definition of parallel machine total tardiness problem. This algorithm is tested using the problem set given by Serifoglu and Ulusoy [14] and the results are compared to their results for the case where the weight of the earliness penalties is zero (In this case their problem also reduces to total tardiness problem).
The next subsection overviews the general TS concept and its application in scheduling theory. Section 2 describes the key aspects of the TS approach used. Numerical experimentation, which compares several alternative approaches and leads towards a robust TS algorithm tailored to solve the problem at hand, is discussed in Section 3. The paper concludes with discussion of results and further studies in Section 4.

1.2 Tabu Search: The Concept and Literature

Tabu Search (Glover and Laguna [16], Reeves [17]) is a metaheuristic that guides a local heuristics search procedure to explore the solution space beyond local optimality. TS allows intelligent problem solving by the incorporation of *adaptive memory* and *responsive exploration*. Key elements of the search path are selectively remembered and strategic choices are made to guide the search into otherwise difficult regions. The adaptive memory usage is a clever compromise between the rigid memory structure of exact techniques like Branch & Bound and the memoryless heuristics like local search procedures.

The basic procedure of TS can be summarized as follows. Starting form an initial solution, TS iteratively moves from the current solution to its best neighbour, even if this new solution is worse than the one available, until a pre-specified stopping criterion becomes true. In order to avoid cycling and becoming trapped in local optima, certain moves that lead to previously explored regions are forbidden or declared *tabu*, forming the short-term memory of TS. The tabu status of a move may be cancelled making it an allowable move if an *aspiration criterion* is satisfied (if, for instance, the tabu move leads to a new best solution). The length of time during which a certain move is classified as tabu, *tabu tenure*, is an important parameter for tabu search. Small values of tabu tenure lead to cycling whereas large values have the risk of prohibiting some good moves. Also, tabu tenures may be adjusted over time, i.e. dynamic tenure, to induce a balance between closely examining one region (intensification) and moving to different paths of the solution space (diversification).

In order to further improve the performance of TS, longer-term strategies like intensification and diversification may be included. Diversification implies forcing the search towards a region that is maximally diverse from the current neighbourhood. It should be mentioned at this point that the diversification strategy of TS is not a randomization process, but rather it uses long-term memory. Intensification tries to lead the search towards solutions whose features are historically found good by modifying neighbour selection moves or by initiating a return to previous "elite" solutions.

A large number of successful applications of TS for scheduling problems can be found in literature. Among these Widmer and Hertz [18], Taillard [19], Kim [20] and Reeves [21] worked on the flowshop scheduling problem. Various TS algorithms for job shop scheduling are presented by Widmer [22], Barnes and Chambers [23], Sun et al. [24], Dell'Amico and Trubian [25] and Valls et. Al. [26]. Laguna et al. [27] study the single machine scheduling problem with the objective of minimizing the sum of setup costs and delay penalties and propose a hybrid neighbourhood. James and Buchanan [28] develop enhanced TS strategies for the single machine early/tardy scheduling problem. Hübscher and Glover [29] apply a candidate list strategy and introduce an influential diversification to parallel machine scheduling to minimize the makespan. Another study, which investigates diversification, is by Laguna, Glover and Kelly [30]. Minimum makespan in a flow shop with parallel machines is presented by Nowicki and Smutnicki [31], who employ a neighbourhood based on blocks of operations on a critical path. A similar block approach is used by Liaw [32] for makespan minimization for an open shop. Park and Kim [33] compare SA and TS for a parallel machine scheduling problem where jobs have equal due dates and equal ready times for minimizing holding costs.

2 DESCRIPTION OF THE TABU SEARCH APPROACH

This section outlines the totally deterministic TS algorithm tailored to the generalized parallel machine total tardiness problem by discussing several of the key concepts such as initial solutions, tabu classifications, candidate list structures, tabu tenure and intensification/diversification strategies.

2.1 Initial Solutions

Two methods are used to generate starting solutions: The first method uses EDD based list scheduling where jobs are ordered with respect to their earliest due dates and then scheduled on the machine that will complete them earliest. The second method is adapted from the KPM heuristic given by Koulamas [3] by incorporating setup times and ready times. This new heuristic is called FPM.

2.2 Neighbourhood Generation

Insert moves and pairwise exchanges (swaps) are two of the frequently used move types in permutation problems. An insert move identifies two particular jobs and places the first job in the location that directly precedes the location of the second job. A swap move, on the other hand, places each job in the location previously occupied by the other, and can be considered as a move that combines two insert moves. In the parallel machine scheduling problem the new locations may be on different machines as well as on the same machine. Swap moves involving jobs on different machines do not cause a change in the number of jobs on machines.

The neighbourhood used in this study has a “hybrid” structure in which the complete “insert neighbourhood” is enlarged by including swap moves for jobs that are on different machines only. Hence, the neighbourhood also includes moves that create different sequences without changing the number of jobs on machines.

2.3 Candidate List Strategies

For situations where the neighbourhood of a solution is large or its elements are expensive to evaluate, candidate list strategies are essential to restrict the number of solutions examined on a given iteration [16]. The purpose of these rules is to screen the neighbourhood so as to concentrate on promising moves at each iteration. When the aggressive nature of TS in selecting the next solution is considered, rules for generating and evaluating good candidates become critical for the efficiency of the search process. Since jobs have distinct ready times, different processing times on different types of machines and sequence dependent setup times, calculation of total tardiness for a given move is a tedious task. Although this is implemented in an efficient way by first determining the affected jobs and updating the tardiness values for only those jobs, move value calculation is still time consuming. Therefore, a good candidate list strategy, which saves time, is critical for the efficiency of the TS algorithm.

In this study three candidate list strategies, which are described below, are tested. Since the neighbourhood generated by swap moves is already smaller than that generated by insert moves, all these strategies are applied only for insert moves.

The Maximum Tardy vs. Maximum Early Approach: In this approach only the jobs on the machine with the highest contribution to total tardiness are chosen as candidates for insert operations to the machine with the highest contribution to total earliness. Since this approach has been shown to be quite fast and decreases the size of the neighbourhood considerably, it is called the '*High Candidate List Strategy*'.

The Maximum Tardy Approach: In this approach the jobs on the machine with the highest contribution to total tardiness are considered for an insert operation on any other machine. Since this approach is slower and has reduced cropping of the neighbourhood as compared to the high Candidate List Strategy, it is called the '*Low Candidate List Strategy*'.

The Ready Time Closeness Approach: In this approach a job is allowed to be inserted only in positions where its starting time remains in a range of its ready time, i.e. ready time ± a threshold value. The threshold value has been determined to be the sum of the maximum processing time and the maximum setup time of all the jobs. This strategy has the purpose of avoiding situations in which jobs are placed in quite unrelated positions causing long idle times and is called the '*Distance Candidate List Strategy*'. It is the fastest among all of the strategies used.

2.4 Tabu Classification

In this study, two alternatives for tabu classifications are used. Tabu Classification 1 (TC1) is position related and therefore has an arc approach, which classifies all schedules where arc ((i-1), i) is included as tabu, where i is the job that moves and (i-1) is its immediate predecessor. Thus TC1 prohibits a recently moved job i from becoming the immediate successor of job i-1 again during tabu duration. This requires all newly added arcs by a move to be checked to see if there is a tabu restriction. Tabu Classification 2 (TC2) on the other hand, is related to the path of the search, restricting certain moves to be repeated within tabu tenure, i.e. inserting i after j or swapping i and j, and requires a smaller number of comparisons.

The last element to be mentioned here is the aspiration criterion, which allows the tabu status of a move to be overridden if it yields a solution better than the best obtained so far.

2.5 Tabu Tenure

Two approaches for tenure selection are employed in this study: using a single tenure value throughout the search versus systematically varying the tenure among a number of values.

For the first approach, the range of tenure values that provides good performance for each problem size are identified and then an empirical rule that depends on the size of the problem instance and yields a fixed tenure value within these ranges is determined.

The systematic dynamic tenure strategy tested in this paper consists of creating a sequence of small (S), medium (M) and large (L) tabu tenure values all in the ranges determined as stated above and repeating this sequence throughout the search. Varying tabu tenure in this manner actually provides a balance between intensification and diversification. Short tabu tenures allow fine-tuning of neighbourhood search and close examination of regions around a local optimum, while long tenures help moving to different parts of the solution space [34].

2.6 Long Term Strategies

Although some intensification and diversification aspects are thus introduced in the TS mechanism through the use of systematic dynamic tenure, these concepts are further investigated by developing some longer term strategies.

The diversification strategy used in this study is different than what is usually employed in literature in that it does not use frequency based information but rather relies on realizing that the search is trapped in some undesirable region (i.e. a deep valley or a large plateau) and forcing it out by resorting to a very large tenure which literally means remembering the whole search history. Thus, after a pre-specified number of non-improving iterations during the normal course of TS, the current tenure is multiplied by a large multiplier and a diversification phase commences. After a specified number of iterations a major disruption is achieved and the short term memory TS is resumed.

The intensification strategy employed consists of keeping a bounded length sequential list of elite solutions during the short-term memory TS and, after erasing all memory, restarting and carrying out a search of a given length from each of these solutions.

3 COMPUTATIONAL STUDIES

A software called "*WinMeta*" is implemented in Visual C++ to conduct the necessary experimentation. WinMeta can be used to generate new problem instances with specific parameters defined in pre-assigned ranges, or to take problems previously created as inputs. The software has embedded in it all the strategies developed in this study. The solution scheme for a problem instance can be specified by the user by selecting a combination of these strategies and providing a set of parameters via a user friendly GUI. WinMeta dynamically produces the total tardiness-total earliness graph of the run, which enables the user to get a clear idea of the topology of the search space. A sample screen of WinMeta is provided in Figure 1. These features of the software allow flexible experiment design and easy tuning of the parameters employed in the TS strategies developed. Taking full advantage of the capabilities of the software developed, extensive experimentation is performed. The experiments are conducted on a Pentium 2-MMX 350 MHz CPU, Host Bus 100 MHz with 128 MB RAM. The problem set and the results obtained are presented in the next sections.

3.1 Example Problems

Although WinMeta can be used to generate a new set of problems, in this study it is preferred to use the benchmark problem set due to Serifoglu and Ulusoy [14]. They propose a genetic crossover operator for parallel machine scheduling with earliness and tardiness penalties. Their problem consists of scheduling a set of independent jobs with sequence dependent setup times, distinct duedates and ready times on a number of parallel machines with different processing rates. Their test problems were generated using the design in Table 1 with 20 instances for each combination. In each problem, it has been assumed that there are two machine types and the machines are evenly distributed among these two types which differ only in their processing rates. The details regarding the generation of the processing times, ready times and duedates can be obtained from the referred paper.

From this set of problems, the 40-job and 60-job problem sets with a maximum set-up duration of 4 time-units are used in our study. The 20-job set is discarded because it turned out to be trivial for the total tardiness measure. Thus the test set used consists of 80 problems. The first two digits in the problem name encoding correspond to the number of jobs, third digit to the number of machines, the fourth to maximum setup length and the last two digits give the instance number. As an example, problem "40247" is the seventh instance of the 40 jobs-2 machines case with a maximum setup duration of 4 time units. The data from the problem sets is scaled up by 100 to avoid decimal numbers.

3.2 Designing the Short-Term Memory TS

The experiments performed at this phase are aimed at designing a short-term memory TS for the problem under consideration by comparing the alternative approaches discussed previously. These experiments are restricted to the first 10 problems of the 60-job set. For all the experiments a stopping criterion of 5000 non-improving iterations is applied.

The first step is to compare the performances of the two tabu classification methods. Each method is employed over a tenure range of [0-250] in increments of 10. For each problem, the percent improvement from the EDD initial solution (i.e. (EDD-Best)/EDD) at each tenure and the average of the percent improvements over the stated range of tenures are computed, and given in Table 2 as Avg. % Impr. The grand averages are given in the last row of Table 2. As clearly seen from the results, Tabu Classification 2 (TC2) gives quite competitive results with those of Tabu Classification 1 (TC1). Considering the fact that TC2 is 10% faster than TC1, TC2 is chosen to be used for the rest of the study.

The second step is the comparison of the two initial solution heuristics: EDD and FPM. The results can be seen in Table 3, where the pairwise comparison represents the percent improvement of EDD-initialized solution over FPM-initialized solution. As seen from these results FPM is dominated by EDD. Therefore EDD is chosen as the default initial solution generation heuristic.

As the third step, the first 10 problems in the 60-job and 40-job sets are tested with tabu tenures starting from 0 to 250, augmenting by 10 at each trial in order to determine a good tabu tenure. This extensive experimentation reveals that no single tenure value can give the best solutions to all problem instances, but good ranges of tenure values can be located. So the efforts are turned to designing an empirical formula depending on problem size that yields an effective tenure for all classes of problems. As a result, $k \times n\sqrt{n}$ turns out to be a reasonable compromise with k = 0.5. However, the same formula performs poorly when applied with a candidate list strategy since a candidate list strategy reduces the neighbourhood size considerably and the tenure value has to be discounted accordingly. After further experimentation, the tenure formula to be used along with a candidate list strategy is determined as $k \times n\sqrt{n}\big/(m-0.5)$, where again, k = 0.5.

The last step is the comparison of the candidate list strategies using this empirical formula. The results are summarized in Table 4 and Table 5. As seen from the results, it can be concluded that the '*Low Candidate List Strategy*' dominates the others.

The short-term memory TS algorithm developed in this section is used in solving all 80 problems in the problem set and the results are presented in the first two columns of Tables 6 to 9. These columns represent the case when there is no candidate list strategy with the case when the candidate list strategy is “low”, respectively. The results indicate that the “low” candidate list strategy is very powerful; not only improving the performance but at the same time dramatically decreasing the CPU time. Moreover, Tables 10 to 13 demonstrate that the short-term TS with the “low” candidate list strategy yields much superior results as compared to the GA solutions [3].

Hence, the TS algorithm with the “low” candidate list strategy is a successful solution method for the parallel machine scheduling problem with sequence dependent setup times. The studies from this point onward will aim to find ways of making efficient use of the computational time saved by applying the candidate list strategy in order to improve the solution further.

3.3 Dynamic Tenure

Based on the observation that problem structure is sensitive to tenure value, a systematic dynamic tenure strategy is also tested. The parameter k in the empirical tenure formula is used in varying the tenure value. A set of different small (S), medium (M) and large (L) tenures as given by different sets of *k* values (k_S, k_M, k_L respectively) and three different cycle patterns of these tenures are tested. These are shown in Table 14. Each tenure is to be applied for [2×medium tenure] iterations. In all experiments the candidate list strategy is “Low” and the stopping criterion is 5000 non-improving iterations. It is concluded that the LMMSMM string with k_S = 0.35, k_M = 0.5 and k_L = 0.8 is the best among these structures.

The results of the dynamic tenure structures tested in this study are presented in Table 15 where a comparison is made between the different structures for the entire set of problems. The solutions of the complete problem set for the selected dynamic tenure structure are presented in the third columns of Tables 6 to 9. The summary of the conclusions is provided in Table 16. It can be said that incorporating the dynamic tenure structure into the base TS algorithm does not improve the solution quality. It does, however, diminish the CPU time significantly.

3.4 Diversification

Three parameters are required to completely define the diversification strategy employed in this study. The first parameter, the phase criterion, defines when to start the diversification phase, and is expressed in number of non-improving iterations that should be completed before concluding that the search has stagnated. The second parameter defines the length of the diversification phase. The last parameter is the tenure multiplier. The tenure value obtained by multiplying the current tenure by its multiplier is used throughout the diversification phase after which the short-term TS resumes with the original tenure.

Extensive experimentation over these parameters was performed and the results of these experiments are provided in Tables 17 to 20 where a comparison between the alternative parameter settings can be made. These results indicate that the best combination of these parameters is to employ the diversification strategy in an overall search duration of 8000 non-improving iterations, where the phase criterion is set at 5000 non-improving iterations, the diversification duration is set at 100 non-improving iterations and the tenure multiplier is chosen to be 1000.

The result of applying this diversification strategy over the base TS algorithm can be seen in column 4 of Tables 6 to 9. The improvement brought to the problems with non-zero solutions through diversification is summarized in Table 21.

3.5 Intensification

In order to fully describe the intensification strategy used in this study the number and nature of the elite solutions to be stored during the short-term memory TS should be specified. Also, the duration of intensification around each local optimum and the search strategy to be used during the intensification phase have to be defined. Experimentation on two different sets of these parameters with and without the dynamic tenure structure is done and the results of these experiments are summarized in Tables 22 to 25. Based on the results of this experimentation it is decided that intensifying around two elite solutions, where an elite solution is defined as a best solution that cannot be improved for at least 300 iterations, provides good results. The intensification phase is adopted after 5000 non-improving iterations of short-term memory TS and lasts for 1500 non-improving iterations around each elite solution.

Interestingly, the best results are obtained when the candidate list strategy is dropped during intensification. This is expected because intensification is a fine-tuning process, and closer examination of the neighbourhood by including the moves previously screened by the candidate list strategy helps in fine-tuning.

In Tables 6 to 9 where the final results are summarized, the intensification column (column 5) uses the above combination of parameters. The improvement brought by intensification over the problems with non-zero solutions is summarized in Table 26.

The best TS results obtained throughout the various stages of experimentation performed in this study are recorded for future reference as benchmark values. These results are reported in Table 27, where each problem and its respective best TS result are seen.

4 CONCLUSIONS

In this paper, a robust TS algorithm for the solution of a very complex the parallel machine scheduling problem where jobs have sequence dependent setup times, distinct duedates and ready times is investigated. The major components of TS are tackled through extensive experimentation and as a result, a completely deterministic TS algorithm is defined. The performance of the algorithm is tested using an existing set of problems from literature, and the obtained results are dramatically better than those that were previously reported.

The most critical TS component in this algorithm is its context related candidate list strategy. The so-called "low" candidate list strategy considers job insertions from the machine with the maximum contribution to total tardiness to each of the other machines respectively. The results reveal that this candidate list strategy is very successful in isolating desirable regions of the neighbourhood, which not only increases the speed of the search, but also improves the solution quality with its power to overcome topological traps and direct the search to good regions.

Generally, the intensification strategy employed performs better than the diversification strategy for these problems. However, both strategies are able to bring some improvement over the short-term memory TS solution within the time frame given by the TS with no candidate list strategy. This therefore has been an efficient way of using the time saved by applying the "low" strategy.

The observation that the improvement brought by the diversification and intensification efforts is limited to less than 1 % can be attributed to the power of the base algorithm arguing that the solutions are already good. But unfortunately it is hard to confirm this since it is not possible to obtain the optimal results. A good lower bound for the parallel machine total tardiness problem with setup times and ready times is not available either. However, since there are 20 non-trivial problems with zero objective value, it can be argued that in at least ¼th of these problems the TS algorithm is able to find the optimal solution.

The diversification and intensification strategies employed in this paper do not use any frequency-information and they are context-independent strategies that can be applied to any problem. It may be an interesting further research to try to incorporate some problem dependent information in creating influential moves towards some elite solutions or away from already searched regions for intensification and diversification purposes, respectively.

5 REFERENCES

[1] Baker KR, Scudder GD. Sequencing with Earliness and Tardiness Penalties: A Review. Operations Research 1990; 38 (1): 22-35.

[2] Cheng TCE, Gupta MC. Survey of Scheduling Research Involving Due Date Determination Decisions. EJOR 1989; 38: 156-166.

[3] Koulamas CP. The Total Tardiness Problem: Review and Extensions. Operations Research 1994; 42: 1025-1041.

[4] Du J, Leung JYT. Minimizing Total Tardiness on One Machine is NP-hard. Mathematics of Operations Research 1990; 15: 483-495.

[5] Root JG. Scheduling with Deadlines and Loss Functions on k Parallel Machines. Management Science 1965; 11: 460-475.

[6] Lawler EL. A 'pseudopolynomial' Algorithm for Sequencing Jobs to Minimize Total Tardiness. Annals of Discrete Mathematics 1977; 1: 331-342.

[7] Elmaghraby SE, Park SH. Scheduling Jobs on a Number of Identical Machines. AIIE Transactions 1974; 6: 1-13.

[8] Dessouky MM. Scheduling Identical Jobs with Unequal Ready Times on Uniform Parallel Machines to Minimize the Maximum Lateness. Computers and Industrial Engineering 1998; 34 (4): 793-806.

[9] Wilkerson LJ, Irwin JD. An Improved Algorithm for Scheduling Independent Tasks. AIIE Transactions 1971; 3: 239-245.

[10] Dogramaci A, Surkis J. Evaluation of a Heuristic for Scheduling Independent Jobs on Parallel Identical Processors. Management Science 1979; 25: 1208-1216.

[11] Ho JC, Chang YL. Heuristics for Minimizing Mean Tardiness for m Parallel Machines. Naval Research Logistics 1991; 38: 367-381.

[12] Koulamas C. Decomposition and Hybrid Simulated Annealing Heuristics for the Parallel-Machine Total Tardiness Problem. Naval Research Logistics 1997; 44: 109-125.

[13] Bean JC. Genetic Algorithms and Random Keys for Sequencing and Optimization. ORSA Journal on Computing 1994; 6: 154-160.

[14] Sivrikaya-Serifoglu F, Ulusoy G. Parallel Machine Scheduling with Earliness and Tardiness Penalties. Computers Operations Research 1999; 26: 773-787.

[15] Balakrishnan N, Kanet JJ, Sridharan SV. Early/Tardy Scheduling with Sequence Dependent Setups on Uniform Parallel Machines. Computers Operations Research 1999; 26: 127-141.

[16] Glover F, Laguna M. Tabu Search. London: Kluwer Academic Publishers, 1997.

[17] Reeves CR. Modern Heuristic Techniques for Combinatorial Problems. New York: John Wiley & Sons, 1993.

[18] Widmer M, Hertz A. A New Heuristic Method for the Flow Shop Sequencing Problem. EJOR 1989; 41: 186-193.

[19] Taillard E. Some Efficient Heuristic Methods for the Flow Shop Sequencing Problem. EJOR 1990; 47: 65-74.

[20] Kim Y-D. Heuristics for flow shop scheduling problems minimizing mean tardiness. JORS 1993; 44: 19-28.

[21] Reeves CR. Improving the Efficiency of Tabu search for Machine Scheduling Problems. JORS 1993; 44: 375-382.

[22] Widmer M. Job Shop Scheduling with Tooling Constraints: A Tabu Search Approach. JORS 1991; 42: 75-82.

[23] Barnes JW, Chambers JB. Solving the Job Shop Scheduling Problem with Tabu Search. IIE Transactions 1995; 27: 257-263.

[24] Sun D, Batta R, Lin L. Effective Job Shop Scheduling Through Active Chain Manipulation. Computers and Operations Research 1995; 22: 159-172.

[25] Dell'Amico M, Trubian M. Applying Tabu Search to the Job-Shop Scheduling Problem. Annals of Operations Research 1993; 41: 231-252.

[26] Valls V, Perez MA, Quintanilla MS. A Tabu Search Approach to Machine Scheduling. EJOR 1998; 106: 277-300.

[27] Laguna M, Barnes JW, Glover F. Tabu Search Methods for a Single Machine Scheduling Problem. Journal of Intelligent Manufacturing 1991; 2: 63-74.

[28] James RJW, Buchanan JT. Performance Enhancements to Tabu Search for the Early/Tardy Scheduling Problem. EJOR 1998; 106: 254-265.

[29] Hübscher R, Glover F. Applying Tabu Search with Influential Diversification to Multiprocessor Scheduling. Computers Operations Research 1994; 21 (8): 877-884.

[30] Kelly JP, Laguna M, Glover F. A Study of Diversification Strategies for the Quadratic Assignment Problem. Computers Operations Research 1994; 21 (8): 885-893.

[31] Nowicki E, Smutnicki C. The Flow Shop with Parallel Machines: A Tabu Search Approach. EJOR 1998; 106: 226-253.

[32] Liaw, Ching-Fang. A Tabu Search Algorithm for the Open Shop Scheduling Problem. Computers Operations Research 1999; 26: 109-126.

[33] Park M-W, Kim Y-D. Search Heuristics for a Parallel Machine Scheduling Problem with Ready Times and Due Dates. Computers and Industrial Engineering 1997; 33 (3-4): 793-796.

[34] Costamagna E, Fanni A, Giacinto G. A Tabu Search Algorithm for the Optimization of Telecommunication Networks. EJOR 1998; 106: 357-372.

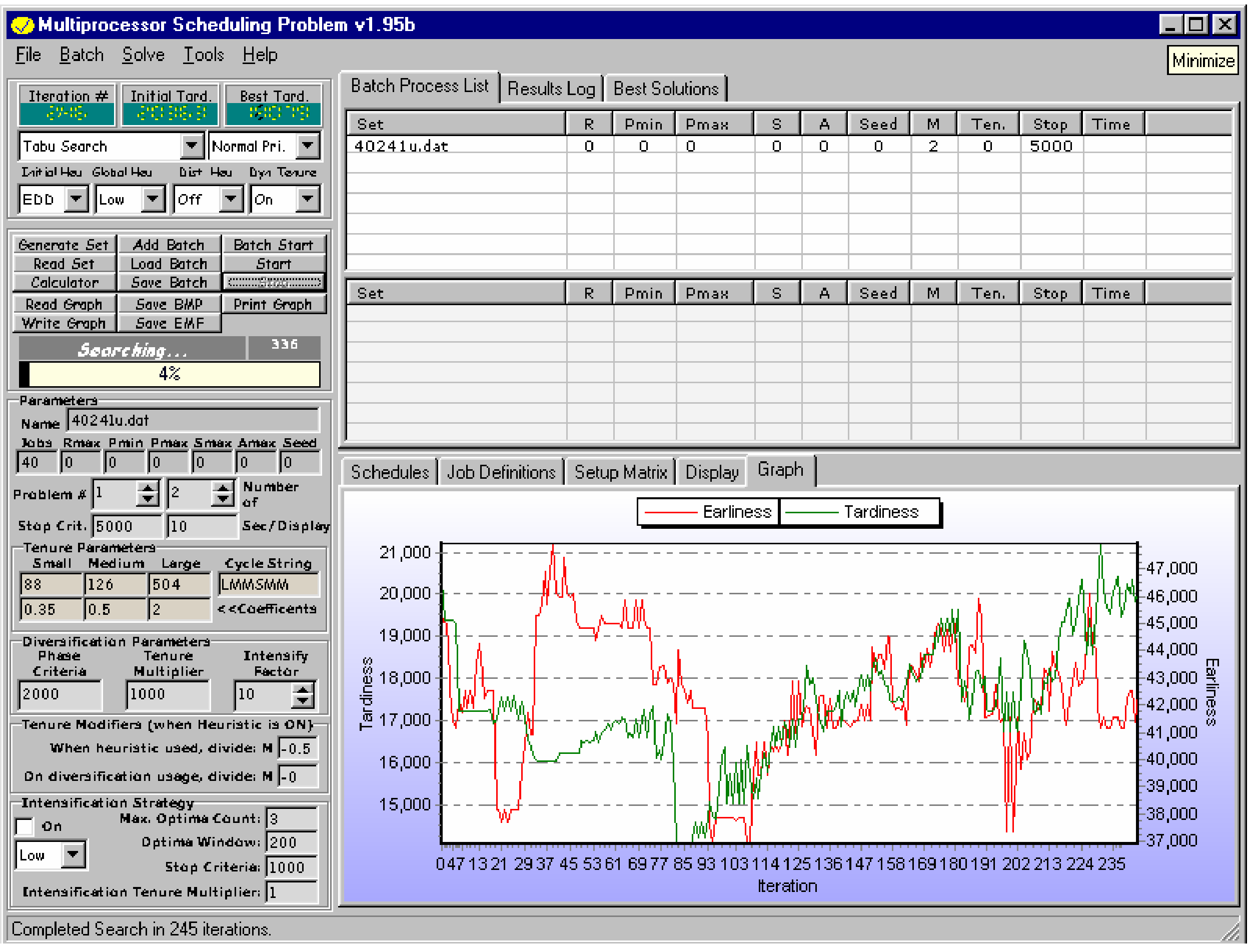

Figure 1 Sample Screen for ‘*WinMeta*’

Table 1 Problem Design Parameters

Number of jobs: n	20, 40, 60
Number of machines: m	2, 4
Maximum setup duration	4,8

Table 2 Comparison of the two Tabu Classification Methods over a tenure range of [0-250]

	TC1	**TC2**
Problem	**Avg.% Impr.**	**Avg.% Impr.**
60241	55.99	51.87
60242	65.61	63.65
60243	34.89	33.83
60244	47.36	61.22
60245	48.48	61.04
60246	50.96	51.51
60247	41.37	41.43
60248	59.41	59.91
60249	59.01	57.93
602410	63.98	64.34
60441	100.00	100.00
60442	66.20	66.11
60443	67.57	63.71
60445	51.78	50.60
60446	70.30	64.27
60447	51.00	51.58
60448	90.20	90.20
60449	95.65	95.44
604410	54.33	54.68
Grand Average	61.79	62.28

Table 3 Effect of the Starting Solution

		EDD		FPM		Pairwise comparison
Problem	**Tenure**	**Initial**	**Best**	**Initial**	**Best**	$\frac{(FPM-EDD)*100}{EDD}$
60241	170	33133	14205	45555	15035	5.843
60242	250	20542	6594	25505	6990	6.005
60243	230	28152	17483	39689	17838	2.031
60244	250	140200	73060	102375	73030	-0.041
60245	230	71646	36005	130395	35508	-1.380
60246	230	110746	50492	87395	53106	5.177
60247	130	46177	26916	50297	27054	0.513
60248	170	20853	8042	42830	8130	1.094
60249	250	43017	16790	55065	17924	6.754
602410	210	62912	21336	83359	22269	4.373
60441	ALL	1297	0	12946	0	0.000
60442	210	14307	3717	23127	4032	8.475
60443	230	5784	350	21811	947	170.571
60444	ALL	0	0	2043	0	0.000
60445	230	6915	2703	12729	3388	25.342
60446	150	2256	635	5744	1087	71.181
60447	190	12188	5354	25284	5006	-6.500
60448	230	1110	0	5102	0	0.000
60449	190	3284	43	11802	0	-100.000
604410	230	17494	5748	32175	5029	-12.509
Average % Difference						9.346

Table 4 Comparison of the Candidate List Strategies-60 jobs, 2 machines

60 JOBS - 2 MACHINES							
	Tenure	**155**		**155**		**155**	
		Candidate List Strategy					
		Low		**High**		**Distance**	
Problem Name	**EDD Initial**	**Best**	**% Impr.**	**Best**	**% Impr.**	**Best**	**% Impr.**
60241	33133	14366	56.64	16936	48.88	14468	56.33
60242	20542	6704	67.36	7352	64.21	7399	63.98
60243	28152	18352	34.81	18994	32.53	18611	33.89
60244	140200	73113	47.85	74754	46.68	77971	44.39
60245	71646	37265	47.99	38492	46.27	39204	45.28
60246	110746	50975	53.97	54162	51.09	53752	51.46
60247	46177	26804	41.95	27682	40.05	28295	38.72
60248	20853	8270	60.34	8738	58.10	8952	57.07
60249	43017	17803	58.61	19823	53.92	18310	57.44
602410	62912	22172	64.76	24923	60.38	22255	64.63
Average % Impr.		**53.43**		**50.21**		**51.32**	

Table 5 Comparison of the Candidate List Strategies-60 jobs, 4 machines

60 JOBS - 4 MACHINES							
	Tenure	**66**		**66**		**66**	
		Candidate List Strategy					
		Low		**High**		**Distance**	
Problem Name	**EDD Initial**	**Best**	**% Impr.**	**Best**	**% Impr.**	**Best**	**% Impr.**
60441	1297	0	100.00	0	100.00	0	100.00
60442	14307	3973	72.23	5997	58.08	5552	61.19
60443	5784	512	91.15	2206	61.86	3517	39.19
60444	0	0	-	0	-	0	-
60445	6915	2961	57.18	3053	55.85	3952	42.85
60446	2256	364	83.87	504	77.66	1262	44.06
60447	12188	5249	56.93	5472	55.10	6640	45.52
60448	1110	0	100.00	0	100.00	0	100.00
60449	3284	43	98.69	436	86.72	196	94.03
604410	17494	4993	34.67	6432	15.84	8937	48.91
Average % Impr.		**77.19**		**67.90**		**63.97**	

Table 6 Final Results for 60 jobs-2 machines

60 JOBS - 2 MACHINES										
k_S-k_M-k_L	0.5		0.5		0.35-0.5-0.8		0.5		0.5	
Cycle String	M		M		LMMSMM		M		M	
Strategies	none		low		low + dynamic		low + diversification		low + intensification	
Non-improving Iterations	5000		5000		5000		8000		8000	
Problem	best	% impr	best	% impr	best	% impr	best	% impr	best	% impr
60241	14205	57.13	14366	56.64	14677	55.70	14360	56.66	14350	56.69
60242	6990	65.97	6704	67.36	6990	65.97	6570	68.02	6662	67.57
60243	18094	35.73	18352	34.81	17749	36.95	17593	37.51	18352	34.81
60244	74054	47.18	73113	47.85	73389	47.65	73113	47.85	73113	47.85
60245	35690	50.19	37265	47.99	35543	50.39	35488	50.47	36406	49.19
60246	53173	51.99	50975	53.97	52825	52.30	50975	53.97	50975	53.97
60247	27152	41.20	26804	41.95	26776	42.01	26804	41.95	26804	41.95
60248	8493	59.27	8270	60.34	8998	56.85	8270	60.34	8087	61.22
60249	17576	59.14	17803	58.61	17254	59.89	17336	59.70	17695	58.87
602410	21577	65.70	22172	64.76	21434	65.93	22172	64.76	21518	65.80
602411	11453	75.78	11694	75.27	11860	74.92	11694	75.27	11334	76.04
602412	14216	53.36	14080	53.81	14991	50.82	14080	53.81	14080	53.81
602413	12806	65.97	13237	64.82	13303	64.65	12978	65.51	13185	64.96
602414	6951	68.52	7069	67.99	6941	68.57	7069	67.99	7048	68.09
602415	20502	44.35	20017	45.67	20068	45.53	20017	45.67	20017	45.67
602416	24281	49.36	24047	49.85	23883	50.19	24047	49.85	24047	49.85
602417	13554	40.57	13877	39.15	12222	46.41	13419	41.16	13874	39.17
602418	40459	60.51	40632	60.34	40237	60.72	40632	60.34	39989	60.97
602419	351	90.61	256	93.15	300	91.98	256	93.15	256	93.15
602420	24544	63.67	24813	63.27	26500	60.78	24813	63.27	23612	65.05
average % improvement		57.31		57.38		57.41		57.86		57.73
average CPU		670.40		490.90		435.05		826.85		761.35

Table 7 Final Results for 60 jobs-4 machines

60 JOBS - 4 MACHINES										
k_S-k_M-k_L	0.5		0.5		0.35-0.5-0.8		0.5		0.5	
Cycle String	M		M		LMMSMM		M		M	
Strategies	none		low		low		low + diversification		low + intensification	
Non-improving Iterations	5000		5000		5000		8000		8000	
Problem	**best**	**% impr**	**best**	**% impr**	**Best**	**% impr**	**best**	**% impr**	**best**	**% impr**
60441	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
60442	3451	75.88	3973	72.23	4006	72.00	3697	74.16	3913	72.65
60443	935	83.83	512	91.15	155	97.32	512	91.15	512	91.15
60444	0	-	0	-	0	-	0	-	0	-
60445	3550	48.66	2961	57.18	2737	60.42	2961	57.18	2737	60.42
60446	828	63.30	364	83.87	364	83.87	364	83.87	364	83.87
60447	5468	55.14	5249	56.93	5064	58.45	4775	60.82	5029	58.74
60448	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
60449	43	98.69	43	98.69	0	100.00	43	98.69	43	98.69
604410	7490	57.19	4993	71.46	6039	65.48	4993	71.46	4975	71.56
604411	4962	56.58	4717	58.73	4937	56.80	4717	58.73	4553	60.16
604412	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
604413	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
604414	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
604415	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
604416	264	92.70	123	96.60	90	97.51	123	96.60	123	96.60
604417	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
604418	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
604419	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
604420	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
average % improvement		**85.89**		**88.78**		**89.04**		**89.09**		**89.15**
average CPU		**265.16**		**116.21**		**94.65**		**167.95**		**199.50**

Table 8 Final Results for 40 jobs-2 machines

40 JOBS - 2 MACHINES										
k_S-k_M-k_L	0.5		0.5		0.35-0.5-0.8		0.5		0.5	
Cycle String	M		M		LMMSMM		M		M	
Strategies	none		low		low		low + diversification		low + intensification	
Non-improving Iterations	5000		5000		5000		8000		8000	
Problem	best	% impr	best	% impr	best	% impr	best	% impr	best	% impr
40241	14079	30.86	14079	30.86	14079	30.86	14079	30.86	14079	30.86
40242	3946	58.25	4013	57.54	3946	58.25	3946	58.25	3946	58.25
40243	3335	62.96	3335	62.96	3335	62.96	3335	62.96	3335	62.96
40244	10758	31.21	10095	35.45	10095	35.45	10095	35.45	10095	35.45
40245	19703	35.13	19748	34.98	19722	35.07	19748	34.98	19703	35.13
40246	26767	51.47	26372	52.18	26372	52.18	26372	52.18	26372	52.18
40247	18565	63.87	18565	63.87	19324	62.39	18565	63.87	18565	63.87
40248	37513	41.35	37658	41.12	37789	40.92	37658	41.12	37658	41.12
40249	1142	87.20	1055	88.18	1055	88.18	1055	88.18	1055	88.18
402410	1270	80.87	1038	84.37	1038	84.37	1038	84.37	1038	84.37
402411	1726	61.52	1835	59.09	1869	58.33	1726	61.52	1726	61.52
402412	8288	46.75	8331	46.47	8465	45.61	8331	46.47	8199	47.32
402413	8382	64.54	8382	64.54	8382	64.54	8382	64.54	8382	64.54
402414	5860	56.36	5869	56.29	5869	56.29	5869	56.29	5860	56.36
402415	21977	53.13	22378	52.28	22134	52.80	22134	52.80	22190	52.68
402416	43502	45.19	43502	45.19	43502	45.19	43502	45.19	43502	45.19
402417	15816	42.00	15816	42.00	15976	41.42	15816	42.00	15816	42.00
402418	6391	55.80	5866	59.43	6430	55.53	5866	59.43	5866	59.43
402419	27258	35.78	27258	35.78	28192	33.58	27258	35.78	27258	35.78
402420	2934	53.03	2934	53.03	2934	53.03	2934	53.03	2934	53.03
Average % improvement		52.86		53.28		52.85		53.46		53.51
Average CPU		213.35		145.15		118.85		224.10		224.70

Table 9 Final Results for 40 jobs-4 machines

40 JOBS - 4 MACHINES										
k_S-k_M-k_L	0.5		0.5		0.35-0.5-0.8		0.5		0.5	
Cycle String	M		M		LMMSMM		M		M	
Strategies	none		low		low		low + diversification		low + intensification	
Non-improving Iterations	5000		5000		5000		8000		8000	
Problem	best	% impr	best	% impr	best	% impr	best	% impr	best	% impr
40441	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40442	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40443	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40444	0	-	0	-	0	-	0	-	0	-
40445	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40446	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40447	1155	78.57	922	82.89	1216	77.44	922	82.89	914	83.04
40448	166	89.88	68	95.85	79	95.18	68	95.85	66	95.98
40449	129	91.62	0	100.00	0	100.00	0	100.00	0	100.00
404410	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
404411	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
404412	0	-	0	-	0	-	0	-	0	-
404413	2807	71.91	2851	71.47	2919	70.79	2851	71.47	2851	71.47
404414	3456	47.21	2704	58.70	2704	58.70	2704	58.70	2704	58.70
404415	1388	77.51	1388	77.51	1886	69.44	1388	77.51	1388	77.51
404416	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
404417	0	-	0	-	0	-	0	-	0	-
404418	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
404419	0	-	0	-	0	-	0	-	0	-
404420	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
Average % improvement		91.04		92.90		91.97		92.90		92.92
Average CPU		57.00		28.45		19.40		36.10		43.60

Table 10 Comparison of GA [3] vs TS for 60 jobs - 2 machines

60 JOBS - 2 MACHINES			
Problem	**GA**	**TS**	**% Improvement of TS over GA**
60241	72860	14366	80.28
60242	74948	6704	91.06
60243	93203	18352	80.31
60244	127175	73113	42.51
60245	110234	37265	66.19
60246	148363	50975	65.64
60247	59213	26804	54.73
60248	69940	8270	88.18
60249	98100	17803	81.85
602410	91911	22172	75.88
602411	58755	11694	80.10
602412	54686	14080	74.25
602413	102444	13237	87.08
602414	88232	7069	91.99
602415	90994	20017	78.00
602416	84974	24047	71.70
602417	37049	13877	62.54
602418	81804	40632	50.33
602419	55911	256	99.54
602420	119553	24813	79.25
Average % Improvement over GA			**75.07**

Table 11 Comparison of GA [3] vs TS for 60 jobs - 4 machines

60 JOBS - 4 MACHINES			
Problem	**GA**	**TS**	**% Improvement of TS over GA**
60441	27626	0	100.00
60442	23326	3973	82.97
60443	40861	512	98.75
60444	18057	0	100.00
60445	13608	2961	78.24
60446	9732	364	96.26
60447	22731	5249	76.91
60448	33076	0	100.00
60449	25279	43	99.83
604410	36781	4993	86.43
604411	42430	4717	88.88
604412	17914	0	100.00
604413	30541	0	100.00
604414	9370	0	100.00
604415	20035	0	100.00
604416	14276	123	99.14
604417	32919	0	100.00
604418	13761	0	100.00
604419	13442	0	100.00
604420	29440	0	100.00
Average % Improvement over GA			**95.37**

Table 12 Comparison of GA [3] vs TS for 40 jobs- 2 machines

40 JOBS - 2 MACHINES			
Problem	**GA**	**TS**	**% Improvement of TS over GA**
40241	25482	14079	44.75
40242	10039	4013	60.03
40243	6224	3335	46.42
40244	17971	10095	43.83
40245	34632	19748	42.98
40246	43730	26372	39.69
40247	35683	18565	47.97
40248	61017	37658	38.28
40249	8951	1055	88.21
402410	11097	1038	90.65
402411	4071	1835	54.93
402412	15907	8331	47.63
402413	24500	8382	65.79
402414	12755	5869	53.99
402415	32672	22378	31.51
402416	56979	43502	23.65
402417	34456	15816	54.10
402418	17006	5866	65.51
402419	35856	27258	23.98
402420	7122	2934	58.80
Average % Improvement over GA			**51.13**

Table 13 Comparison of GA [3] vs TS for 40 jobs - 4 machines

40 JOBS - 4 MACHINES			
Problem	**GA**	**TS**	**% Improvement of TS over GA**
40441	2980	0	100.00
40442	4259	0	100.00
40443	2002	0	100.00
40444	2422	0	100.00
40445	131	0	100.00
40446	5549	0	100.00
40447	6348	922	85.48
40448	5745	68	98.82
40449	3304	0	100.00
404410	4270	0	100.00
404411	2142	0	100.00
404412	726	0	100.00
404413	12067	2851	76.37
404414	9821	2704	72.47
404415	7812	1388	82.23
404416	0	0	-
404417	2244	0	100.00
404418	3766	0	100.00
404419	581	0	100.00
404420	6008	0	100.00
Average % Improvement over GA			**95.55**

Table 14 Tested dynamic tenure structures

Set	k_S	k_M	k_L	Cycle String
1	0.4	0.5	0.7	SMLM
2	0.35	0.5	0.8	SMLM
3	0.35	0.5	1	SMLM
4	0.35	0.5	0.8	LMMSMM
5	0.35	0.5	0.8	MMSMML
6	0.35	0.5	2	LMMSMM
7	0.35	0.5	2	MMSMML

Table 15 Comparison of Different Dynamic Tenure Implementations

k_S-k_M-k_L	0.4-0.5-0.7		0.35-0.5-0.8		0.35-0.5-1		0.35-0.5-2		0.35-0.5-2	
Cycle string	SMLM		SMLM		SMLM		MMSMML		LMMSMM	
Candidate List Strategy	low		low		low		low		low	
Problem	Best	%Impr	Best	%Impr	Best	%Impr	Best	%Impr	Best	%Impr
40241	14079	30.86	14079	30.86	14079	30.86	14079	30.86	14079	30.86
40242	3946	58.25	3946	58.25	3946	58.25	3946	58.25	3946	58.25
40243	3335	62.96	3335	62.96	3335	62.96	3335	62.96	3335	62.96
40244	10095	35.45	10095	35.45	10095	35.45	10095	35.45	10095	35.45
40245	19722	35.07	19722	35.07	19722	35.07	19703	35.13	19703	35.13
40246	26543	51.87	26372	52.20	26543	51.90	26990	51.06	26868	51.28
40247	18585	63.83	18585	63.80	18565	63.90	18585	63.83	18565	63.87
40248	37610	41.20	37610	41.20	37710	41.00	38068	40.48	37513	41.35
40249	1055	88.18	1055	88.20	1055	88.20	1055	88.18	1055	88.18
402410	1270	80.87	1038	84.40	1109	83.30	1038	84.37	1353	79.62
40441	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40442	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40443	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40444	0	-	0	-	0	-	0	-	0	-
40445	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40446	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
40447	1145	78.75	947	82.40	1194	77.80	1034	80.81	1076	80.03
40448	131	92.01	79	95.20	160	90.20	78	95.24	116	92.93
40449	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
404410	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
60241	14350	56.69	14677	55.70	14531	56.14	14531	56.14	14677	55.70
60242	6990	65.97	7059	65.64	7399	63.98	7399	63.98	7129	65.30
60243	18135	35.58	18460	34.43	17824	36.69	17824	36.69	17607	37.46
60244	73292	47.72	73171	47.81	73633	47.48	73633	47.48	73426	47.63
60245	36484	49.08	35486	50.47	37959	47.02	37959	47.02	36351	49.26
60246	53494	51.70	53204	51.96	50374	54.51	50374	54.51	50529	54.37
60247	27232	41.03	27232	41.03	27087	41.34	27087	41.34	26804	41.95
60248	8834	57.64	9774	53.13	9030	56.70	9030	56.70	8060	61.35
60249	17747	58.74	17312	59.76	17618	59.04	17618	59.04	17339	59.69
602410	22398	64.40	20943	66.71	21824	65.31	21824	65.31	21899	65.19
60441	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
60442	4115	71.24	2737	80.87	4307	69.90	4307	69.90	3451	75.88
60443	620	89.28	273	95.28	441	92.38	441	92.38	727	87.43
60445	2827	59.12	3625	47.58	2773	59.90	2773	59.90	2777	59.84
60446	436	80.67	364	83.87	380	83.16	380	83.16	399	82.31
60447	5029	58.74	5029	58.74	5249	56.93	5249	56.93	4773	60.84
60448	0	100.00	0	100.00	0	100.00	0	100.00	0	100.00
60449	43	98.69	0	100.00	43	98.69	43	98.69	43	98.69
604410	5858	23.35	4893	35.98	5273	31.01	5273	31.01	4981	71.53
Avg.%Impr.	69.10		69.97		69.45		69.65		70.90	

Table 16 Dynamic Tenure Structure Results

Comparison Criterion	% Improvement		No. of Better Solutions Found		CPU	
k_S-k_M-k_L	**0.5**	**0.35-0.5-0.8**	**0.5**	**0.35-0.5-0.8**	**0.5**	**0.35-0.5-0.8**
Cycle String	**M**	**LMMSMM**	**M**	**LMMSMM**	**M**	**LMMSMM**
Strategies	**low**	**low+dynamic**	**low**	**low+dynamic**	**low**	**low+dynamic**
Non-improving Iterations	**5000**	**5000**	**5000**	**5000**	**5000**	**5000**
40/2	53.28	52.85	7	3	145.15	118.85
40/4	92.90	91.97	4	0	28.45	19.4
60/2	57.38	57.41	11	9	490.9	435.05
60/4	88.78	89.04	3	5	116.21	94.65

Table 17 Comparison of Diversification Structures for 60 jobs-2 machines

(tenure multiplier = 1000)

	60 JOBS - 2 MACHINES						
	k_S-k_M-k_L	0.5		0.5		0.5	
	Cycle String	M		M		M	
	Strategies	low + diversification		low + diversification		low + diversification	
	Phase Criterion	2000		5000		3000	
	Non-improving Iterations	5000		8000		7500	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
60241	33133	14205	57.13	14360	56.66	14350	56.69
60242	20542	7032	65.77	6570	68.02	6797	66.91
60243	28152	17772	36.87	17593	37.51	17296	38.56
60244	140200	73483	47.59	73113	47.85	73113	47.85
60245	71646	35457	50.51	35488	50.47	36130	49.57
60246	110746	52918	52.22	50975	53.97	50975	53.97
60247	46177	26679	42.22	26804	41.95	26804	41.95
60248	20853	8702	58.27	8270	60.34	8270	60.34
60249	43017	17907	58.37	17336	59.70	17862	58.48
602410	62912	22354	64.47	22172	64.76	21756	65.42
602411	47295	11204	76.31	11694	75.27	11694	75.27
602412	30482	14216	53.36	14080	53.81	14080	53.81
602413	37630	13103	65.18	12978	65.51	13237	64.82
602414	22084	7142	67.66	7069	67.99	6948	68.54
602415	36844	20502	44.35	20017	45.67	20032	45.63
602416	47951	24606	48.69	24047	49.85	24281	49.36
602417	22807	13709	39.89	13419	41.16	12304	46.05
602418	102449	39000	61.93	40632	60.34	39116	61.82
602419	3739	555	85.16	256	93.15	256	93.15
602420	67564	26193	61.23	24813	63.27	25197	62.71
Average % improvement			**56.86**		**57.86**		**58.05**
Average CPU			**431.80**		**826.85**		**662.20**

Table 18 Comparison of Diversification Structures for 60 jobs-4 machines

(tenure multiplier = 1000)

	60 JOBS - 4 MACHINES						
	k_S-k_M-k_L	0.5		0.5		0.5	
	Cycle String	M		M		M	
	Strategies	low + diversification		low + diversification		low + diversification	
	Phase Criterion	2000		5000		3000	
	Non-improving Iterations	5000		8000		7500	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
60441	1297	0	100.00	0	100.00	0	100.00
60442	14307	3254	77.26	3697	74.16	3973	72.23
60443	5784	350	93.95	512	91.15	768	86.72
60444	0	0	-	0	-	0	-
60445	6915	2713	60.77	2961	57.18	2961	57.18
60446	2256	364	83.87	364	83.87	364	83.87
60447	12188	4772	60.85	4775	60.82	5191	57.41
60448	1110	0	100.00	0	100.00	0	100.00
60449	3284	43	98.69	43	98.69	0	100.00
604410	17494	5512	68.49	4993	71.46	4993	71.46
604411	11429	5010	56.16	4717	58.73	4717	58.73
604412	2114	0	100.00	0	100.00	0	100.00
604413	1410	0	100.00	0	100.00	0	100.00
604414	1815	0	100.00	0	100.00	0	100.00
604415	230	0	100.00	0	100.00	0	100.00
604416	3614	266	92.64	123	96.60	58	98.40
604417	2344	0	100.00	0	100.00	0	100.00
604418	277	0	100.00	0	100.00	0	100.00
604419	83	0	100.00	0	100.00	0	100.00
604420	4770	0	100.00	0	100.00	0	100.00
Average % improvement			**89.09**		**89.09**		**88.74**
Average CPU			**142.84**		**167.95**		**161.65**

Table 19 Comparison of Diversification Structures for 40 jobs-2 machines

(tenure multiplier = 1000)

	40 JOBS - 2 MACHINES						
	k_S-k_M-k_L	0.5		0.5		0.5	
	Cycle String	M		M		M	
	Strategies	low + diversification		low + diversification		low + diversification	
	Phase Criterion	2000		5000		3000	
	Non-improving Iterations	5000		8000		7500	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
40241	20363	14079	30.86	14079	30.86	14079	30.86
40242	9452	3946	58.25	3946	58.25	3946	58.25
40243	9003	3335	62.96	3335	62.96	3335	62.96
40244	15640	10095	35.45	10095	35.45	10095	35.45
40245	30372	19748	34.98	19748	34.98	19748	34.98
40246	55152	27362	50.39	26372	52.18	26372	52.18
40247	51380	18585	63.83	18565	63.87	18565	63.87
40248	63959	37718	41.03	37658	41.12	37789	40.92
40249	8925	1055	88.18	1055	88.18	1055	88.18
402410	6640	1270	80.87	1038	84.37	1038	84.37
402411	4485	1777	60.38	1726	61.52	1726	61.52
402412	15563	8288	46.75	8331	46.47	8331	46.47
402413	23639	8382	64.54	8382	64.54	8382	64.54
402414	13427	5869	56.29	5869	56.29	5869	56.29
402415	46894	22125	52.82	22134	52.80	22378	52.28
402416	79365	43502	45.19	43502	45.19	43502	45.19
402417	27271	15901	41.69	15816	42.00	15816	42.00
402418	14459	5983	58.62	5866	59.43	5866	59.43
402419	42442	28192	33.58	27258	35.78	27258	35.78
402420	6246	2989	52.15	2934	53.03	2934	53.03
Average % improvement			**52.94**		**53.46**		**53.43**
Average CPU			**150.50**		**224.10**		**204.15**

Table 20 Comparison of Diversification Structures for 40 jobs-4 machines

(tenure multiplier = 1000)

	40 JOBS – 4 MACHINES						
	k_S-k_M-k_L	0.5		0.5		0.5	
	Cycle String	M		M		M	
	Strategies	low + diversification		low + diversification		low + diversification	
	Phase Criterion	2000		5000		3000	
	Non-improving Iterations	5000		8000		7500	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
40441	1638	0	100.00	0	100.00	0	100.00
40442	206	0	100.00	0	100.00	0	100.00
40443	207	0	100.00	0	100.00	0	100.00
40444	0	0	-	0	-	0	-
40445	124	0	100.00	0	100.00	0	100.00
40446	607	0	100.00	0	100.00	0	100.00
40447	5389	1056	80.40	922	82.89	1100	79.59
40448	1640	48	97.07	68	95.85	68	95.85
40449	1539	0	100.00	0	100.00	0	100.00
404410	821	0	100.00	0	100.00	0	100.00
404411	665	0	100.00	0	100.00	0	100.00
404412	0	0	-	0	-	0	-
404413	9993	3071	69.27	2851	71.47	3201	67.97
404414	6547	2704	58.70	2704	58.70	2704	58.70
404415	6171	1445	76.58	1388	77.51	1388	77.51
404416	123	0	100.00	0	100.00	0	100.00
404417	0	0	-	0	-	0	-
404418	963	0	100.00	0	100.00	0	100.00
404419	0	0	-	0	-	0	-
404420	420	0	100.00	0	100.00	0	100.00
Average % improvement			**92.63**		**92.90**		**92.48**
Average CPU			**28.94**		**36.10**		**28.90**

Table 21 Diversification Strategy Results

Comparison Criterion	% Improvement		CPU		No. of non-zero solutions	No. of Better Solutions Found by Diversification
k_S-k_M-k_L	0.5	0.5	0.5	0.5		
Cycle String	M	M	M	M		
Strategies	low	low+ diversification	low	low+ diversification		
Non-improving Iterations	5000	8000	5000	8000		
40/2	53.28	53.46	145.15	224.10	20	3
40/4	92.90	77.28	28.45	36.10	5	0
60/2	57.38	57.86	490.90	826.85	20	7
60/4	88.78	76.96	116.21	167.95	9	2

Table 22 Comparison of Intensification Structures for 60 jobs-2 machines

	60 JOBS - 2 MACHINES								
	Strategies	low+ intensification		low + dynamic tenure +intensification		low+ intensification		low + dynamic tenure +intensification	
	Intensification Strategy	none		none		none		none	
	Number of Local Optima	3		3		2		2	
	Optima Window	200		200		300		300	
	Intensification Stopping Criterion	1000		1000		1500		1500	
	k_S-k_M-k_L	0.5		0.35-0.5-2		0.5		0.35-0.5-2	
	Cycle String	M		LMMSMM		M		LMMSMM	
	Intensification Factor	1		1		1		1	
	Non-improving Iterations	5000		5000		5000		5000	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
60241	33133	14677	55.70	14350	56.69	14350	56.69	14667	55.73
60242	20542	7129	65.30	6662	67.57	6662	67.57	7129	65.30
60243	28152	17607	37.46	18341	34.85	18352	34.81	17607	37.46
60244	140200	73009	47.93	73113	47.85	73113	47.85	73009	47.93
60245	71646	36119	49.59	36851	48.57	36406	49.19	36119	49.59
60246	110746	50529	54.37	50975	53.97	50975	53.97	50529	54.37
60247	46177	26804	41.95	26804	41.95	26804	41.95	26804	41.95
60248	20853	8060	61.35	8087	61.22	8087	61.22	8060	61.35
60249	43017	17339	59.69	17695	58.87	17695	58.87	17339	59.69
602410	62912	21832	65.30	21518	65.80	21518	65.80	21832	65.30
602411	47295	11204	76.31	11334	76.04	11334	76.04	11204	76.31
602412	30482	14376	52.84	14080	53.81	14080	53.81	14376	52.84
602413	37630	12978	65.51	13185	64.96	13185	64.96	12978	65.51
602414	22084	7212	67.34	7048	68.09	7048	68.09	7212	67.34
602415	36844	20293	44.92	20017	45.67	20017	45.67	20293	44.92
602416	47951	23883	50.19	24047	49.85	24047	49.85	23883	50.19
602417	22807	12259	46.25	13874	39.17	13874	39.17	12259	46.25
602418	102449	39875	61.08	39989	60.97	39989	60.97	39875	61.08
602419	3739	666	82.19	256	93.15	256	93.15	666	82.19
602420	67564	24454	63.81	23612	65.05	23612	65.05	24454	63.81
Average % improvement			**57.45**		**57.70**		**57.73**		**57.46**
Average CPU			**683.75**		**775.60**		**761.35**		**722.30**

Table 23 Comparison of Intensification Structures for 60 jobs-4 machines

	60 JOBS - 4 MACHINES								
	Strategies	low+ intensification		low + dynamic tenure +intensification		low+ intensification		low + dynamic tenure +intensification	
	Intensification Strategy	none		none		none		none	
	Number of Local Optima	3		3		2		2	
	Optima Window	200		200		300		300	
	Intensification Stopping Criterion	1000		1000		1500		1500	
	k_S-k_M-k_L	0.5		0.35-0.5-2		0.5		0.35-0.5-2	
	Cycle String	M		LMMSMM		M		LMMSMM	
	Intensification Factor	1		1		1		1	
	Non-improving Iterations	5000		5000		5000		5000	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
60441	1297	0	100.00	0	100.00	0	100.00	0	100.00
60442	14307	3451	75.88	3913	72.65	3913	72.65	3451	75.88
60443	5784	727	87.43	512	91.15	512	91.15	727	87.43
60444	0	0	-	0	-	0	-	0	-
60445	6915	2777	59.84	2777	59.84	2737	60.42	2777	59.84
60446	2256	399	82.31	364	83.87	364	83.87	399	82.31
60447	12188	4744	61.08	5029	58.74	5029	58.74	4744	61.08
60448	1110	0	100.00	0	100.00	0	100.00	0	100.00
60449	3284	0	100.00	43	98.69	43	98.69	0	100.00
604410	17494	4947	71.72	4975	71.56	4975	71.56	4981	71.53
604411	11429	4755	58.40	4706	58.82	4553	60.16	4700	58.88
604412	2114	0	100.00	0	100.00	0	100.00	0	100.00
604413	1410	0	100.00	0	100.00	0	100.00	0	100.00
604414	1815	0	100.00	0	100.00	0	100.00	0	100.00
604415	230	0	100.00	0	100.00	0	100.00	0	100.00
604416	3614	215	94.05	123	96.60	123	96.60	357	90.12
604417	2344	0	100.00	0	100.00	0	100.00	0	100.00
604418	277	0	100.00	0	100.00	0	100.00	0	100.00
604419	83	0	100.00	0	100.00	0	100.00	0	100.00
604420	4770	0	100.00	0	100.00	0	100.00	0	100.00
Average % improvement			**88.98**		**89.05**		**89.15**		**88.79**
Average CPU			**192.50**		**190.85**		**199.50**		**191.50**

Table 24 Comparison of Intensification Structures for 40 jobs-2 machines

	40 JOBS - 2 MACHINES								
	Strategies	low+ intensification		low + dynamic tenure +intensification		low+ intensification		low + dynamic tenure +intensification	
	Intensification Strategy	none		none		none		none	
	Number of Local Optima	3		3		2		2	
	Optima Window	200		200		300		300	
	Intensification Stopping Criterion	1000		1000		1500		1500	
	k_S-k_M-k_L	0.5		0.35-0.5-2		0.5		0.35-0.5-2	
	Cycle String	M		LMMSMM		M		LMMSMM	
	Intensification Factor	1		1		1		1	
	Non-improving Iterations	5000		5000		5000		5000	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
40241	20363	14079	30.86	14079	30.86	14079	30.86	14079	30.86
40242	9452	3946	58.25	3946	58.25	3946	58.25	3946	58.25
40243	9003	3335	62.96	3335	62.96	3335	62.96	3335	62.96
40244	15640	10095	35.45	10095	35.45	10095	35.45	10095	35.45
40245	30372	19748	34.98	19703	35.13	19703	35.13	19703	35.13
40246	55152	26372	52.18	26807	51.39	26372	52.18	26807	51.39
40247	51380	18565	63.87	18565	63.87	18565	63.87	18565	63.87
40248	63959	37658	41.12	37513	41.35	37658	41.12	37513	41.35
40249	8925	1055	88.18	1055	88.18	1055	88.18	1055	88.18
402410	6640	1038	84.37	1038	84.37	1038	84.37	1038	84.37
402411	4485	1726	61.52	1726	61.52	1726	61.52	1726	61.52
402412	15563	8199	47.32	8288	46.75	8199	47.32	8288	46.75
402413	23639	8382	64.54	8382	64.54	8382	64.54	8382	64.54
402414	13427	5869	56.29	5955	55.65	5860	56.36	5955	55.65
402415	46894	22190	52.68	21712	53.70	22190	52.68	21712	53.70
402416	79365	43502	45.19	43502	45.19	43502	45.19	43502	45.19
402417	27271	15816	42.00	15976	41.42	15816	42.00	15976	41.42
402418	14459	5866	59.43	6019	58.37	5866	59.43	6019	58.37
402419	42442	27258	35.78	27258	35.78	27258	35.78	27258	35.78
402420	6246	2934	53.03	2934	53.03	2934	53.03	2934	53.03
Average % improvement			**53.50**		**53.39**		**53.51**		**53.39**
Average CPU			**218.40**		**183.00**		**224.70**		**194.35**

Table 25 Comparison of Intensification Structures for 40 jobs-4 machines

	40 JOBS - 4 MACHINES								
	Strategies	**low+ intensification**		**low + dynamic tenure +intensification**		**low+ intensification**		**low + dynamic tenure +intensification**	
	Intensification Strategy	**none**		**none**		**none**		**none**	
	Number of Local Optima	**3**		**3**		**2**		**2**	
	Optima Window	**200**		**200**		**300**		**300**	
	Intensification Stopping Criterion	**1000**		**1000**		**1500**		**1500**	
	k_S-k_M-k_L	**0.5**		**0.35-0.5-2**		**0.5**		**0.35-0.5-2**	
	Cycle String	**M**		**LMMSMM**		**M**		**LMMSMM**	
	Intensification Factor	**1**		**1**		**1**		**1**	
	Non-improving Iterations	**5000**		**5000**		**5000**		**5000**	
Problem	**EDD Initial**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**	**best**	**% impr**
40441	1638	0	100.00	0	100.00	0	100.00	0	100.00
40442	206	0	100.00	0	100.00	0	100.00	0	100.00
40443	207	0	100.00	0	100.00	0	100.00	0	100.00
40444	0	0	-	0	-	0	-	0	-
40445	124	0	100.00	0	100.00	0	100.00	0	100.00
40446	607	0	100.00	0	100.00	0	100.00	0	100.00
40447	5389	914	83.04	914	83.04	914	83.04	914	83.04
40448	1640	116	92.93	58	96.46	66	95.98	116	92.93
40449	1539	0	100.00	0	100.00	0	100.00	0	100.00
404410	821	0	100.00	0	100.00	0	100.00	0	100.00
404411	665	0	100.00	0	100.00	0	100.00	0	100.00
404412	0	0	-	0	-	0	-	0	-
404413	9993	3035	69.63	2851	71.47	2851	71.47	3035	69.63
404414	6547	2704	58.70	2704	58.70	2704	58.70	2704	58.70
404415	6171	1445	76.58	1388	77.51	1388	77.51	1445	76.58
404416	123	0	100.00	0	100.00	0	100.00	0	100.00
404417	0	0	-	0	-	0	-	0	-
404418	963	0	100.00	0	100.00	0	100.00	0	100.00
404419	0	0	-	0	-	0	-	0	-
404420	420	0	100.00	0	100.00	0	100.00	0	100.00
Average % improvement			**92.55**		**92.95**		**92.92**		**92.55**
Average CPU			**29.15**		**53.50**		**43.60**		**29.75**

Table 26 Intensification Strategy Results

Comparison Criterion	% Improvement		CPU		No. of non-zero solutions	No. of Better Solutions Found by Intensification
k_S-k_M-k_L	0.5	0.5	0.5	0.5		
Cycle String	M	M	M	M		
Strategies	low	low+ intensification	low	low+ intensification		
Non-improving Iterations	5000	8000	5000	8000		
40/2	53.28	53.46	145.15	224.70	20	5
40/4	92.90	77.28	28.45	43.60	5	2
60/2	57.38	57.86	490.90	761.35	20	12
60/4	88.78	76.96	116.21	199.50	9	5

Table 27 Best Tabu Search Results

Problem	Best Value	Problem	Best Value	Problem	Best Value	Problem	Best Value
40241	14079	**40441**	0	**60241**	14205	**60441**	0
40242	3946	**40442**	0	**60242**	6528	**60442**	2737
40243	3335	**40443**	0	**60243**	17296	**60443**	155
40244	10095	**40444**	0	**60244**	72406	**60444**	0
40245	19695	**40445**	0	**60245**	34640	**60445**	2591
40246	26372	**40446**	0	**60246**	50492	**60446**	339
40247	18565	**40447**	914	**60247**	26660	**60447**	4744
40248	37513	**40448**	48	**60248**	8042	**60448**	0
40249	1055	**40449**	0	**60249**	16790	**60449**	0
402410	1038	**404410**	0	**602410**	20943	**604410**	4626
402411	1726	**404411**	0	**602411**	11204	**604411**	4423
402412	8199	**404412**	0	**602412**	14080	**604412**	0
402413	8382	**404413**	2807	**602413**	12806	**604413**	0
402414	5860	**404414**	2704	**602414**	6874	**604414**	0
402415	21712	**404415**	1388	**602415**	20017	**604415**	0
402416	43502	**404416**	0	**602416**	23883	**604416**	58
402417	15816	**404417**	0	**602417**	12222	**604417**	0
402418	5866	**404418**	0	**602418**	38948	**604418**	0
402419	27258	**404419**	0	**602419**	164	**604419**	0
402420	2934	**404420**	0	**602420**	23514	**604420**	0