Pergamon

Computers Ops Res. Vol. 25, No. 7/8, pp. 625–635, 1998
© 1998 Elsevier Science Ltd. All rights reserved
Printed in Great Britain
PII: S0305-0548(97)00097-X
0305-0548/98 $19.00 + 0.00

A COMPOSITE HEURISTIC FOR THE SINGLE MACHINE EARLY/TARDY JOB SCHEDULING PROBLEM

Maria Teresa Almeida†‡ and Mário Centeno§
Instituto Superior de Economia e Gestão, Universidade Técnica de Lisboa, Rua Miguel Lúpi 20, 1200 Lisboa, Portugal

(Received October 1996; in revised form October 1997)

Scope and Purpose—The single machine early/tardy job scheduling problem (SMETP) consists of determining the best processing sequence for a set of jobs in order to minimize total costs. Minimizing both earliness and tardiness costs pushes the completion of each job to as close to its due date as possible in accordance with the just-in-time philosophy of production planning. As the problem is NP-hard and local optimality conditions may be well away from global optimality conditions new methods are needed to generate near optimal solutions. We propose a new heuristic for the SMETP that alternates search techniques performed at different neighborhood ranges using its own results to guide the alternation in a dynamic way. The computational results we obtained on randomly generated test problems compared very favorably with the results obtained with tabu search and simulated annealing algorithms.

Abstract—The single machine early/tardy job scheduling problem (SMETP) is a NP-hard problem for which most properties of optimal solutions for single machine problems with regular objective function do not hold. In this paper we present a new composite heuristic for the SMETP that combines tabu search, simulated annealing and steepest descent techniques to generate near optimal schedules. Computational experience is reported for a set of randomly generated test problems. © 1998 Elsevier Science Ltd. All rights reserved

Key words: Job scheduling, tabu search, simulated annealing

1. INTRODUCTION

The problem considered in this paper is that of generating the schedule that minimizes the weighted sum of total earliness and total tardiness of a set of n jobs to be processed in a single machine. The problem will be referred to as SMETP (for single machine early/tardy problem). Using the traditional four-parameter classification for scheduling problems [8], the SMETP is a $n/1//ET$ problem.

Single machine problems may be part of more general scheduling problems or may arise on their own representing real world settings in which only one facility is involved or in which one machine acts as a bottleneck for the whole system (see French [8], Morton and Pentico [25]). For years most research focused on performance measures that do not include earliness penalties, such as maximum tardiness, weighted number of late jobs or total weighted completion time (for a review see Lawler *et al.* [22]). However during the last decade the interest on early/tardy problems has been growing stimulated by new management trends such as just-in-time. Earliness costs may represent finished goods inventory maintenance and insurance costs, deterioration or obsolescence costs, opportunity costs due to improductive investment or a combination of them.

The class of $n/1//ET$ problems is a vast class that includes problems with a common due date and problems with distinct due dates, problems with endogenous due dates and problems with given due dates, problems with linear and problems with non linear performance measures, etc.

†To whom all correspondence should be addressed. E-mail: talmeida@iseg.utl.pt.
‡Maria Teresa Almeida is a professor in the Mathematics Department at the ISEG. She earned a Ph.D. in Operational Research at the London School of Economics.
§Mário Centeno earned a M.Sc. in Mathematics for Economics and Business at the ISEG. He was a lecturer in the Mathematics Department at the ISEG and is now a Ph.D. candidate in Economics at Harvard University.

Some of them are polynomially solvable and some other are known to be NP-hard. Hall and Posner [16] studied the minimization of weighted deviation of completion times when the common due date is not early enough to constrain the schedule and Hall *et al.* [15] studied the same problem when the common due date is restrictive. Both are NP-hard problems. Kahlbacker [19] showed that some of the properties valid for these problems still hold for equal slack due date rule problems. When the common due date is considered an endogenous variable the problem is polynomially solvable (Quaddus [28], Chand and Chhajed [6]). When different jobs may have distinct due dates, minimizing total deviation costs is a polynomially solvable problem if due dates are treated as endogenous variables (Baker and Scudder [3]). When the due dates are distinct and given, minimizing total deviation costs is NP-hard (Garey *et al.* [9]), but minimizing maximum lateness is polynomially solvable (Sidney [30], Lakshinrayan *et al.* [21]).

Although so many variations may seem of little practical importance they in fact reflect real world diversity of settings. In all cases the main goal is to push the completion of jobs to as close to their due dates as possible to avoid both earliness and tardiness costs. The mathematical formulation is established in each case according to the practice of the setting it is intended to model. For a survey on early/tardy problems and their applications see Baker and Scudder [3].

In Section 2 we describe the SMETP and review previous research on it. Section 3 is devoted to the aspects of neighborhood search needed to the presentation made of the composite heuristic in Section 4. In Section 5 we report the computational experience and in Section 6 we make the final remarks.

2. THE SINGLE MACHINE EARLY/TARDY PROBLEM

The single machine early/tardy job scheduling problem addressed in this paper may be described as follows. Let $N = \{1, 2, \ldots, n\}$ be a set of n jobs to be processed in a single machine. Each job i ($i \in N$) is available at time zero, requires p_i units of time to be processed, has a given due date d_i and shall be processed without interruption. The machine can only process one job at a time and shall not be idle, i.e. shall process all jobs during the time window $[0, T]$, $T = \sum_{i=1}^{n} p_i$. If c_i denotes the completion time for job i then $e_i = \max\{0, d_i - c_i\}$ is its earliness and $t_i = \max\{0, c_i - d_i\}$ is its tardiness. If a nonnegative penalty a_i is defined for each unit of time job i ($i \in N$) is early and a nonnegative penalty b_i is defined for each unit of time job i ($i \in N$) is late then the total cost to be minimized may be stated as $\sum_{i=1}^{n} (a_i e_i + b_i t_i)$. This is a non-regular function, i.e., a function that does not monotonically increase with job completion times. Most properties valid for regular objective function problems do not hold when the objective is non-regular, making the search for optimal schedules much harder. Some non-regular problems are known to have V-shaped optimal schedules but this is not the case for the SMETP.

Garey *et al.* [9] proved that the SMETP with $a_i = b_i = 1$ ($i \in N$) is NP-hard so the more general nonnegative penalty case considered here is also NP-hard.

The condition that the machine shall not be idle was addressed by Davis and Kanet [7]. They developed an efficient procedure that allocates idle time to form an optimal schedule for a given job permutation and embedded it in an enumerative search algorithm over the finite domain of job permutations. From their computational experience they concluded that what differentiated optimum solutions with and without idle time allowed was not so much the existence of idle time in between jobs but the fact that the sequences were themselves different, as most frequently idleness occurred before the first job was processed.

Davis and Kanet [7] also investigated the influence of the due date range factor on the results and concluded it is quite strong. As pointed out by Ow and Morton [27] when job due dates are close there are multiple "clashes" between jobs making local optimality very different from global optimality. These instances are therefore particularly hard to solve.

The relative importance of the earliness and tardiness penalties is strongly related to the tardiness factor, a coarse measure of the proportion of jobs that might be expected to be tardy. Two special cases of the E/T problem were shown by Ow and Morton [27] to be easy to solve: those for which the weighted shortest processing time sequence results in a schedule with no early jobs and those for which the weighted longest processing time sequence results in a schedule with no tardy jobs.

The magnitude of the processing times determines the value of T, the total time required to process the whole set of jobs. Given a due date range and a tardiness factor this magnitude does not seem to influence significantly the results for constructive and neighborhood search heuristics, Centeno [4].

Algorithms for the SMETP were presented in Abdul-Razaq and Potts [2], Ow and Morton [27] and Sousa and Wolsey [31]. Abdul-Razaq and Potts [2] solved to optimality problems with up to 25 jobs using a branch and bound algorithm with lower bounds derived from state-space relaxations of a dynamic programming formulation. Ow and Morton [27] developed two priority rules (a linear priority rule and an exponential priority rule) to guide the generation of near optimal sequences and developed a method based on the beam search technique. Centeno and Almeida [5] presented computational results, for problems with up to 50 jobs, for deterministic and randomized versions of several constructing heuristics: Ow and Mortons priority rule algorithms, an alternative priority definition and a modified version of their filtered beam search algorithm. Sousa and Wolsey [31] proposed a time indexed binary integer programming formulation and a cutting plane/branch and bound algorithm for a class of four different single machine scheduling problems, including the SMETP. They report results for three 20 job and two 30 job SMETPs.

3. NEIGHBORHOOD SEARCH

The idea of using neighborhood search for finding near optimal solutions to combinatorial problems dates back to the sixties (Lin [23]). In the initial versions the search was performed following a monotonic path, i.e., changing a known solution for a new one only if the change yields an improvement in the objective function value. Although in some cases these methods (known as descent methods in the OR literature and as hill climbing in the AI literature) may be quite efficient they have the drawback of frequently getting trapped into poor quality local optima. To overcome this drawback new search strategies have been proposed, namely simulated annealing, Kirkpatrick *et al.* [20], and tabu search, Glover [10], [11].

Simulated annealing is a randomized search technique that owes its name to the analogy with the procedure for growing crystals by heating the material past the melting point and then cooling it slowly. Non-improving solutions may be accepted, with a set probability that decreases along the procedure. Tabu search, in its original version, is a deterministic search strategy that also allows, under certain circumstances, changes to non-improving solutions. To avoid getting trapped by cycling (due to the reversal of previous changes) it forbids some changes, making them tabu. Both techniques are now well known and will not be fully described here. For a general review see Reeves [29].

Simulated annealing has been applied to many combinatorial problems including several job scheduling problems (e.g. Ogbu and Smith [26], Van Laarhoven *et al.* [32]). As illustrated by the extensive computational experiments reported in Johnson *et al.* [17, 18], the results are heavily dependent on parameter tuning and on the structure of the solution space to be searched. It is therefore difficult to draw general conclusions about its performance. It seems, however, that the computing time it requires to provide good solutions is often excessive.

Tabu search has been the focus of intensive research as illustrated by the 23 papers included in Glover *et al.* [14] (7 of them under the heading "Applications in Production and Scheduling"). The efficiency of Glovers initial basic version has been improved by new developments and extensions. For a review on important refinements see Glover *et al.* [12]. The results of tabu search are also parameter tuning dependent and, again, difficult to assess as a whole. It seems that it is, generally speaking, less time demanding than simulated annealing.

General suggestions to build hybrid algorithms are given in Glover and Laguna [13] among others. To solve single machine problems with non-regular objective functions Mittenthal *et al.* [24] proposed a two-phase algorithm that generates the initial feasible solution with a greedy heuristic (phase 1) and then applies simulated annealing search to find a near-optimal solution (phase 2). Their algorithm applies only when optimal schedules are known to be V-shaped, which is not the case for the SMETP. Adenso-Diaz [1] proposed an algorithm for the scheduling tardiness problem that first generates a feasible solution using a dispatching rule and then applies simulated annealing search and tabu search in sequence. Simulated annealing is used as

a modifier of the initial solution in an attempt to save iterations in the tabu search. The cardinality of the neighborhoods in the tabu search is decreased along the procedure to reduce computing time.

The composite algorithm we propose combines three search techniques, associated with different neighborhood definitions, in an alternating four-phase procedure. The order in which phases are alternated is not established in advance. It is dynamically guided by the insight gained over the feasible region along the process. A detailed presentation of the algorithm is made in the next section.

4. THE COMPOSITE HEURISTIC

All search techniques we use require the definition of a neighborhood to limit the search to. By a neighborhood we mean here a set of schedules that may be generated from the current one performing simple operations. For single machine problems such simple operations (usually referred to as moves in the tabu search terminology) are, traditionally, swaps and reinsertions. A swap is an operation involving two jobs whose positions in the schedule are exchanged. A reinsertion consists of removing a single job from its current position in the schedule and inserting it into a new one. Based on these two simple operations we define three neighborhoods for a given schedule, say S:

$$\mathcal{N}_1(S) = \{S' \text{: } S' \text{ is obtained swapping a pair of jobs adjacent in } S\}$$

$$\mathcal{N}_2(S) = \{S' \text{: } S' \text{ is obtained swapping a pair of jobs that are sequenced at most } q_1 \text{ positions apart in } S\}$$

where q_1 is an integer parameter set to a small positive value.

$$\mathcal{N}_3(S) = \{S' \text{: } S' \text{ is obtained reinserting one job at least } q_2 \text{ positions apart its position in S}\}$$

where q_2 is an integer parameter.

The impact of a move depends on the neighborhood considered.

For a n-job problem the cost function to be minimized has $2n$ terms (see Section 2). A move within $\mathcal{N}_1(S)$ alters the value of at most 4 terms only (those associated with the pair of swapped jobs). A move within $\mathcal{N}_2(S)$ may alter the values of up to $2q_1+4$ terms as, if the processing times for the swapped jobs are not equal, both the earliness and the tardiness penalties of the jobs scheduled in between them may be changed. A move within $\mathcal{N}_3(S)$ may alter the value of up to $2q + 2$ ($q \geq q_2$), q being the number of jobs scheduled in between the old and the new position of the job subjected to reinsertion. We set q_2 to q_1+1.

Based on the impact in each case we define three search ranges: short range (neighborhood $\mathcal{N}_1$), medium range (neighborhood $\mathcal{N}_2$) and long range (neighborhood $\mathcal{N}_3$).

In our composite heuristic different neighborhood search techniques are alternated, performed at different ranges, in a four-phase iterative procedure. The search alternation is guided by the insight obtained over the sub-regions explored in previous phases. The procedure is repeated until the number of consecutive iterations without global improvement reaches a set limit. The main goal of our approach is to enhance the search to be able to get a good look through the feasible region within a reasonable amount of computing time. Roughly, each phase of the procedure may be characterized as follows. Phase 1 is an intensive deterministic search phase. Phase 2 corresponds to a restricted randomized diversification that, when successful, is followed by a 2-optimal deterministic optimization (Phase 3). Phase 4 may be viewed as a random diversification that is tried when the ability to find a better solution in the subset of the feasible region under investigation seems to be exhausted.

The algorithm may be summarized as follows:

Initialization

$x\_\text{now} \leftarrow$ initial feasible solution
$c(x\_\text{now}) \leftarrow$ cost of the initial feasible solution

max_iter ← maximum number of iterations allowed without improving the best known solution
x_best = x_now {*best solution known so far*}
$c(x\_best) = c(x\_now)$ {*cost of the best solution known so far*}
no_sa = TRUE {*no simulated annealing was started at the best solution known so far*}
iter = 0 {*number of iterations since x_best was updated for the last time*}

Iteration

Repeat while iter < max_iter

Phase 1 (tabu search/short range)
iter = iter + 1
Perform tabu search, from x_now, with neighborhood definition $\mathcal{N}_1$.
Update x_now and $c(x\_now)$ every time a new solution is selected along the search.
Whenever $c(x\_now) < c(x\_best)$ set:
  x_best = x_now
  $c(x\_best) = c(x\_now)$
  iter = 0
  no_sa = TRUE
Terminate the search according to a convergence based termination criterion.
If no_sa = TRUE go to phase 2. Otherwise go to phase 4.

Phase 2 (simulated annealing/medium range)
no_sa = FALSE
Perform simulated annealing search, from x_best, with neighborhood definition $\mathcal{N}_2$.
Update x_now and $c(x\_now)$ every time a new solution is accepted.
Whenever $c(x\_now) < c(x\_best)$ set:
  x_best = x_now
  $c(x\_best) = c(x\_now)$
  iter = 0
  no_sa = TRUE
Terminate the search according to a convergence based termination criterion.
If no_sa = TRUE go to phase 3. Otherwise go to phase 4.

Phase 3 (descent search/short range)
Perform descent search, from x_best, to find a $\mathcal{N}_1$-local optimum, x*_local.
If $c(x^*\_local) < c(x\_best)$ set:
  x_best = x*_local
  $c(x\_best) = c(x^*\_local)$
  iter = 0
  no_sa = TRUE
Go to phase 4.

Phase 4 (random selection/long range)
Randomly select x_now in $\mathcal{N}_3$ (x_best).
If $c(x\_now) < c(x\_best)$ set:
  x_best = x_now
  $c(x\_best) = c(x\_now)$
  iter = 0
  no_sa = TRUE
Go to phase 1.

As global optimal solutions for the SMETP do not have any known characteristic shape and local optimality conditions may be well away from global optimality conditions, we are not able to identify in advance "good" sub-regions and "poor" sub-regions within the feasible region. On the other hand, as the SMETP is a permutation problem we cannot resort to the technique

of temporarily dropping some constraints to have an easier access to new sets of solutions. We were thus led to think that a search method for the SMETP should be able to use its own results dynamically to combine balanced search of the whole feasible region with the ability not to miss at least some of the best solutions in each sub-region examined.

Performing simulated annealing search at medium range (phase 2) after completing an intensive tabu search (phase 1) allows for a wider knowledge over the sub-region under examination before it is abandoned (phase 4). If the medium range flexible search does not yield a new best solution the decision to move the search to another sub-region is taken immediately. If a new best solution is found during phase 2 the decision to leave the sub-region is postponed until the search for a local optimum is completed (phase 3).

The general design of our composite heuristic was based on our intuition that alternating the search range and strategies would yield a better coverage of the whole set of feasible solutions than just performing intensive search at short range around particular solutions selected by a diversification step. When a new best solution is found by the medium range more flexible simulated annealing search it did not seem sensible to us to drive the search away before performing some extra short range search around it. With phase 3 we try not to miss good solutions by an untimely diversification decision.

For the implementation of our method it is necessary to set the value for several parameters: the parameters required by the simulated annealing and by the tabu search routines and the parameters required by the termination criterion and the medium range definition. Our implementation is fully described in Section 5.

5. COMPUTATIONAL EXPERIENCE

In this section we present the rules we used to generate the test problems, the details of our implementation of the composite heuristic, the computational results and the criteria we adopted to evaluated them.

5.1. Data generation

To generate the test problems we followed Abdul-Razaq and Potts [2], Ow and Morton [27] and Sousa and Wolsey [31]. We generated 336 problems: 112 with 10 jobs, 112 with 20 jobs and 112 with 50 jobs. For each dimension n, i.e. number of jobs, we generated 8 sets, of 14 problems each, as follows.

The processing times p_i ($i \in N$) were uniformly generated over the integers $\{1, 2, \ldots, 10\}$.

For the due dates d_i ($i \in N$) we considered τ (the tardiness factor) and ρ (the due date range) in the set $\{0.2, 0.4, 0.6, 0.8, 1.0\}$ and generated the due dates uniformly over the integers

$$\{T^*(1-\tau-0.5\rho), \ldots, T^*(1-\tau+0.5\rho)\}$$

for the 14 feasible combinations of values for the pair (τ,ρ).

For the earliness coefficients a_i ($i \in N$) and the tardiness coefficients b_i ($i \in N$) we considered eight schemes:

$$(S1)\ a_i = A_i{*}p_i \quad b_i = B_i{*}p_i$$

$$(S2)\ a_i = 0.75{*}b_i \quad b_i = B_i{*}p_i$$

$$(S3)\ a_i = 0.50{*}b_i \quad b_i = B_i{*}p_i$$

$$(S4)\ a_i = 0.25{*}b_i \quad b_i = B_i{*}p_i$$

with A_i and B_i uniformly distributed over the integers $\{1, 2, \ldots, 10\}$

($S5$) a_i and b_i uniformly distributed over the integers $\{1, 2, \ldots, 10\}$.

($S6$) $a_i = 0.75{*}b_i$, b_i uniformly distributed over the integers $\{1, 2, \ldots, 10\}$.

($S7$) $a_i = 0.50{*}b_i$, b_i uniformly distributed over the integers $\{1, 2, \ldots, 10\}$.

($S8$) $a_i = 0.25{*}b_i$, b_i uniformly distributed over the integers $\{1, 2, \ldots, 10\}$.

In (*S*1)–(*S*4), for each job, the earliness and the tardiness coefficients are related to the processing time. These schemes model situations in which the earliness and the tardiness coefficients reflect the value-added cost. In (*S*2)–(*S*4) and in (*S*6)–(*S*8), for each job, the earliness coefficient is a set proportion of the tardiness coefficient.

5.2. Composite heuristic implementation

As mentioned in Section 3 the results of the search techniques we use are heavily dependent on parameter tuning and it is always possible to argue about the values adopted. Devoting a lot of effort to parameter tuning would make the computing times presented quite misleading, as most of the computing time would be hidden. So we made limited experiments for data generated according to scheme (*S*1) above and adopted the same values for the whole study. This seemed to us the best way to get results whose relative quality would not be significantly affected by different quality parameter tuning. As our main goal was to assess the ability of our composite heuristic to enhance the search when compared with other neighborhood search techniques we implemented the tabu search and the simulated annealing following the guidance given in Glover *et al*. [14] and Johnson *et al*. [17] respectively.

For the tabu search (phase 1) we made both the tabu list length and the maximum number of iterations dependent on the instance dimension, following the general practice. Increasing the tabu length with the number of jobs attempts to achieve a good trade-off between avoiding cycling and disregarding too many solutions in the neighborhood. The maximum number of iterations allowed was also set as an increasing function of the number of jobs to take into account the exponential growth of the solution set cardinality.

For the simulated annealing (phase 2) the neighborhood size was set as an increasing function of the number of jobs again to account for the solution set cardinality but the maximum number of failures allowed had to be decreased with the number of jobs to prevent unacceptable computational times.

The numerical values adopted in each case are given below.

Initialization. The initial feasible solution was generated with Ow and Mortons linear priority rule algorithm, based on our previous experience, Centeno and Almeida [5].

Max_iter was made equal to n (the number of jobs) in each case.

Phase 1. The moves labelled as tabu were those that reversed a swap performed during the last L iterations, with L equal to 3, 5 and 7 for problems with 10, 20 and 50 jobs, respectively. For globally improving moves (i.e., moves that give a new x_best solution) the tabu status was always overrun. The search was stopped after performing n iterations without global improvement, unless this limit was hit after performing a sequence of k locally improving moves, in which case the stopping criterion was overrun until a non-improving move was faced. For n equal to 10, 20 and 50, k was set to 3, 6 and 10, respectively.

Phase 2. We followed closely Johnson *et al*. [17]. The temperature reduction was made according to a multiplying scheme with a factor of 0.95. The procedure was stopped when a minimum threshold of 2% of accepted solutions failed at k temperature levels, after the last global improvement. For n equal to 10, 20 and 50, k was set to 3, 2 and 1, to prevent excessive computational times.

For the definition of neighborhood $\mathcal{N}_2$ we set q_1 equal to 2, 4, 9 for n equal to 10, 20 and 50, respectively.

Phase 3. In the 2-opt search we used the first-best rule, i.e., every time an improving solution was found in the neighborhood it was immediately used to update the current solution, x_now.

5.3. Evaluation criteria

To assess the ability of our algorithm to enhance the search we compared the results given by the composite heuristic, for the test problems, with the results of applying simulated annealing and tabu search, each on its own, to the same test problems. To make the comparison as fair as possible we used the implementation described above for the composite heuristic, except for a few differences to be explained next.

To assess the quality of the solution in each case, we tried to compute the gap to a lower bound on the optimal value, to be described below.

Simulated annealing. We implemented two versions of simulated annealing: one with neighborhood $\mathcal{N}_1$ (ANNEAL1) and one with neighborhood $\mathcal{N}_2$ (ANNEAL2) as in phase 2 of the composite heuristic. To compensate for the extra search opportunities allowed to the composite heuristic we quadruplicated the number of trials at each temperature level.

Tabu search. The implementation used to run tabu search on its own includes a diversification step copied from phase 4 of the composite heuristic. Again to compensate for the extra search opportunities given to the composite heuristic, we augmented 50% the maximum number of diversifications without global improvement and the number of nonimproving moves allowed before a new diversification step is made.

Lower bound. To get an upper bound on the deviation from the optimal solution value, in each case, we tried to compute the gap to the lower bound value provided by the linear relaxation of Sousa and Wolseys [31] integer formulation defined as:

$$\text{GAP} = \left(\frac{v_{\text{H}} - v_{\text{LP}}}{v_{\text{LP}}} \right) \times 100\%$$

where v_{H} is the value obtained with heuristic H and is v_{LP} the optimum value of the LP relaxation.

Their formulation has $O(n + T)$ constraints and $O(nT)$ binary variables. The variables, x_{it}, establish for each job i ($i = 1, \ldots, n$) the moment t ($t = 0, \ldots, T - p_i$) in which it starts being processed. Even restricting the processing times to the set $\{1, \ldots, 10\}$ we had to use a mainframe IBM risc 600 computer to solve the linear problems with the XMP code. As the LP relaxation solutions are highly degenerate, for the 50 job problems we were able to find the LP optimum only in 36.6% of the cases, spending several hours of CPU time to get each solution.

5.4. Computational results

All algorithms were coded in Pascal and were run on an IBM PC Compatible with a 386 INTEL processor.

The results are presented with problems grouped by generating scheme for each dimension n (number of jobs). So, except for some lines of Table 1, each line contains the average results for

Table 1. Average gap to the LP relaxation lower bound (%)

Number of jobs (n)	Generation scheme[a]	Number of problems for which LP lower bound is known	Average gap to the LP lower bound[b]			
			anneal 1	anneal 2	tabu search	comp. heuristic
10	$S1$	14	0.25	0.47	0.19	0.19
	$S2$	14	0.26	0.50	0.18	0.18
	$S3$	14	15.70	0.61	3.64	1.73
	$S4$	14	9.60	2.28	1.94	2.30
	$S5$	14	15.82	3.26	1.74	1.18
	$S6$	14	6.04	2.56	3.35	2.12
	$S7$	14	4.11	4.38	1.81	1.66
	$S8$	14	4.44	6.41	4.20	3.87
20	$S1$	14	3.88	3.70	1.75	1.23
	$S2$	13	6.68	1.70	1.41	2.20
	$S3$	14	6.58	2.70	1.29	0.88
	$S4$	14	6.65	11.11	2.21	2.78
	$S5$	14	5.52	8.78	4.00	4.18
	$S6$	14	3.21	8.36	3.26	3.04
	$S7$	14	16.64	16.09	5.78	4.79
	$S8$	13	10.14	17.60	5.38	1.87
50	$S1$	5	3.74	5.17	1.57	2.90
	$S2$	6	10.89	11.94	5.22	3.65
	$S3$	2	11.05	7.05	5.32	1.70
	$S4$	2	34.82	35.91	5.50	7.91
	$S5$	7	5.13	10.21	4.34	3.46
	$S6$	8	20.19	23.05	4.19	4.76
	$S7$	6	9.69	17.80	5.44	6.99
	$S8$	5	22.70	42.68	12.90	12.21

[a]as described in Section 5.1.
[b]as defined in Section 5.3.

Table 2. Average gap to the best heuristic solution value (%)

		Anneal 1		Anneal 2		Tabu Search		Comp. heuristic	
Number of jobs (n)	generation scheme[a]	average gap to the best heuristic solution value	number of times the heuristic found the best solution	average gap to the best heuristic solution value	number of times the heuristic found the best solution	average gap to the best heuristic solution value	number of times the heuristic found the best solution	average gap to the best heuristic solution value	number of times the heuristic found the best solution
	$S1$	0.06	12	0.28	1	0.00	14	0.00	14
	$S2$	0.07	12	0.30	1	0.00	14	0.00	14
	$S3$	15.00	6	0.06	6	3.07	8	1.18	10
10	$S4$	7.70	8	0.51	3	0.17	13	0.52	9
	$S5$	14.77	10	2.28	0	0.76	12	0.21	13
	$S6$	4.54	9	1.13	0	1.89	12	0.70	12
	$S7$	2.57	11	2.79	0	0.27	13	0.12	13
	$S8$	1.20	11	3.12	1	0.93	12	0.66	12
	$S1$	2.72	6	2.54	0	0.61	9	0.10	13
	$S2$	5.53	3	0.96	2	0.67	11	1.41	9
	$S3$	5.71	7	1.85	2	0.47	9	0.06	12
20	$S4$	4.93	8	9.40	0	0.65	8	1.18	7
	$S5$	2.78	10	6.04	2	1.34	11	1.55	13
	$S6$	0.57	11	5.70	3	0.61	10	0.40	10
	$S7$	11.49	6	10.80	1	1.15	10	0.17	10
	$S8$	7.57	8	15.52	0	3.23	10	0.11	11
	$S1$	2.43	2	2.59	0	0.81	7	0.58	6
	$S2$	9.00	1	9.87	0	1.88	5	0.21	9
	$S3$	11.12	1	10.27	1	2.94	6	0.93	8
50	$S4$	12.43	3	13.12	0	0.98	5	0.85	6
	$S5$	2.91	3	6.19	0	1.15	8	0.12	7
	$S6$	12.32	3	14.80	0	0.59	7	1.33	6
	$S7$	6.28	1	11.40	1	0.56	6	1.68	9
	$S8$	13.28	3	24.28	0	0.60	8	1.02	5

[a]as described in Section 5.1.

14 problems with n jobs (n = 10, 20 or 50 as stated in column (1)) generated by scheme S_i (i = 1, 2... or 8 as stated in column (2)).

Table 1 presents the average gap to the LP relaxation lower bound on the optimum value for each heuristic. For problems with 50 jobs we were able to obtain the LP bound only in 41 out of the 112 cases and for the 20 job problems we failed two. So the average refers to the number of problems indicated in column (3).

The overall average gap is 9.9% for the simulated annealing algorithm with neighborhood $\mathcal{N}_1$ and 7.6% with neighborhood $\mathcal{N}_2$, 3.1% for the tabu search algorithm and 2.6% for the composite heuristic. The largest average gap is 12.21% for the composite heuristic, 12.90% for the tabu search heuristic 34.82% for the simulated annealing algorithm with neighborhood $\mathcal{N}_1$ and 42.68% with neighborhood $\mathcal{N}_2$.

As we were not able to compute the LP relaxation optimum for every problem, Table 2 presents the average deviation from the best heuristic solution and the number of times each algorithm found the best solution.

The best solution was obtained by the simulated annealing algorithm for 155 problems with neighborhood $\mathcal{N}_1$ and for 24 problems with neighborhood $\mathcal{N}_2$. Tabu search provided the best solution for 228 problems and the composite heuristic provided it for 238 problems. The largest average gap for simulated annealing is 15.00% with neighborhood $\mathcal{N}_1$ and 24.28% with neighborhood $\mathcal{N}_2$. For tabu search the largest average gap is 3.23% and for the composite heuristic the largest average gap is only 1.68%.

Table 3 presents the average computing times.

The average CPU time is 32.9 s for the composite heuristic and 60.4 s for the tabu search algorithm. The simulated annealing algorithm has an overall average of 71.7 s with neighborhood $\mathcal{N}_1$ and 45.9 s with neighborhood $\mathcal{N}_2$.

With neighborhood $\mathcal{N}_1$ the simulated annealing algorithm takes longer than the tabu search algorithm and the quality of the solutions it generates is clearly inferior. With neighborhood $\mathcal{N}_2$ the average computing time for simulated annealing is reduced to about 75% of tabu searche's but the solution quality is again clearly inferior.

Table 3. Average CPU time (s)

		Average CPU time (s)			
Number of jobs (n)	generation scheme[a]	anneal 1	anneal 2	tabu search	comp. heuristic
10	$S1$	15.37	11.11	3.50	2.57
	$S2$	15.40	10.09	3.42	2.46
	$S3$	16.25	9.20	5.01	3.63
	$S4$	16.50	11.71	4.45	3.36
	$S5$	19.10	7.78	4.02	7.73
	$S6$	18.90	9.40	4.29	3.90
	$S7$	18.60	7.63	4.43	3.64
	$S8$	21.76	7.37	4.55	4.00
20	$S1$	35.47	62.16	17.47	13.80
	$S2$	26.62	64.25	20.59	10.61
	$S3$	26.60	63.96	21.45	12.25
	$S4$	30.28	61.33	21.20	15.74
	$S5$	33.86	61.31	16.02	21.41
	$S6$	35.60	60.57	14.38	11.15
	$S7$	34.43	60.67	21.04	23.05
	$S8$	36.60	60.19	23.32	16.46
50	$S1$	155.24	70.62	143.97	66.60
	$S2$	145.51	73.08	153.62	81.94
	$S3$	137.62	69.88	120.04	93.98
	$S4$	145.84	66.61	133.67	103.78
	$S5$	183.39	64.52	178.68	45.40
	$S6$	168.77	63.60	186.57	89.53
	$S7$	168.93	63.73	156.59	58.96
	$S8$	213.13	61.31	188.50	93.90

[a]as described in Section 5.1.

The average CPU time for the composite heuristic is only 54% of the average CPU time for the tabu search algorithm and the average gap to the LP lower bound is 16% smaller.

These results seem to indicate that alternating the search strategy, guided by the insight gained along the procedure, may be an efficient method to reduce the computing time and even to improve slightly the solution quality.

As mentioned earlier a very limited parameter tuning was performed (only for problems generated according to scheme $S1$). We believe that all methods performance may be improved but we do not have any reason to believe that the relative performance would be altered.

6. CONCLUDING REMARKS

Most properties of optimal solutions for single machine problems with regular objective function do not hold for the SMETP. This fact makes the search for its optimal solution particularly hard.

In this paper we have proposed a new composite heuristic for the SMETP that combines steepest descent, simulated annealing and tabu search techniques and that does not rely on particular properties of the optimal solution.

From our experience with the SMETP we believe that the good results obtained with the composite heuristic are due to its distinctive feature: the iterative alternation of the search patterns combined with the alternation of search ranges, guided by the insight progressively gained over the sub-regions already explored. We intend now to apply this methodology to other difficult combinatorial problems to further investigate its ability to generate good solutions within reasonable computing times.

Acknowledgements—We are most grateful to our colleague Raul Bras for his assistance in finding the LP relaxation lower bounds and to the two anonymous referees whose comments were most helpful to us.

REFERENCES

1. Adenso-Diaz, B., An SA/Ts mixture algorithm for the scheduling tardiness problem. *European Journal of Operational Research*, 1996, **88,** 516–524.
2. Abdul-Razaq, T. S. and Potts, C. N., Dynamic programming state-space relaxation for single-machine scheduling. *Journal of the Operational Research Society*, 1988, **39,** 141–152.
3. Baker, K. R. and Scudder, G. D., Sequencing with earliness and tardiness penalties: A review. *Operations Research*, 1990, **38,** 22–36.

4. Centeno, M., Métodos Heurísticos para o Problema de Escalonamento de Tarefas numa Máquina com Custos de Antecipação e Atraso, M.Sc. Dissertation, Instituto Superior de Economia e Gestão-Universidade Técnica de Lisboa, Lisboa, 1993.
5. Centeno, M. and Almeida, M. T., Soluções Aproximadas para um Problema de Escalonamento numa Máquina: Algoritmos construtivos. Working-Paper CEMAPRE 2/94, Instituto Superior de Economia e Gestão-Universidade Técnica de Lisboa, 1994.
6. Chand, S. and Chhajed, D., A single machine model for determination of optimal due dates and sequence. *Operations Research*, 1992, **40,** 596–602.
7. Davis, J. S. and Kanet, J. J., Single machine scheduling with early and tardy completion costs. Working-Paper, Clemson University, 1988.
8. French, S., *Sequencing and scheduling, an introduction to the mathematics of the job-shop*, Ellis Horwood, 1990.
9. Garey, M. R., Tarjan, R. E. and Wilfong, G. T., One-processor scheduling with symmetric earliness and tardiness penalties. *Mathematics of Operations Research*, 1988, **13,** 330–348.
10. Glover, F., Tabu search – Part I. *ORSA Journal on Computing*, 1989, **1,** 190–206.
11. Glover, F., Tabu search – Part II. *ORSA Journal on Computing*, 1990, **2,** 4–32.
12. Glover, F., Taillard, E. and de Werra, D., A users guide to tabu search. *Annals of Operations Research*, 1993, **41,** 3–28.
13. Glover, F. and Laguna, M., Tabu search. In *Modern Heuristic Techniques for Combinatorial Problems*, ed. C. Reeves, Blackwell Scientific Publications, 1993.
14. Glover, F., Laguna, M., Taillard, E. and de Werra, D. (eds.), Tabu search, *Annals of Operations Research* 41, J. C. Baltzer AG, 1993.
15. Hall, N. G., Kubiak, W. and Sethi, S. P., Earliness–tardiness scheduling problems, II: Deviation of completion times about a restrictive common due date. *Operations Research*, 1991, **39,** 847–856.
16. Hall, N. G. and Posner, M. E., Earliness–tardiness scheduling problems, I: Weighted deviation of completion times about a common due date. *Operations Research*, 1991, **39,** 836–846.
17. Johnson, D. S., Aragon, C. R., McGeoch, L. A. and Schevon, C., Optimization by simulated annealing: An experimental evaluation; part I graph partitioning. *Operations Research*, 1989, **37,** 865–892.
18. Johnson, D. S., Aragon, C. R., McGeoch, L. A. and Schevon, C., Optimization by simulated annealing: An experimental evaluation; part II graph coloring and number partitioning. *Operations Research*, 1991, **39,** 378–406.
19. Kahlbacher, H. G., Scheduling with monotonous earliness and tardiness penalties. *European Journal of Operational Research*, 1993, **64,** 258–277.
20. Kirkpatrick, S., Gelatt, C. D., Jr. and Vecchi., M., Optimization by simulated annealing. *Science*, 1983, **220,** 671–680.
21. Lakshminarayan, S., Lakshmanan, R., Papineau, R. L. and Rochette, R., Optimal single-machine scheduling with earliness and tardiness penalties. *Operations Research*, 1978, **26,** 1079–1082.
22. Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G. and Shmoys, D. B., Sequencing and scheduling: Algorithms and complexity. In *Logistics of Production and Inventory, Handbooks in Operations Research and Management Science* 4, North-Holland, 1993.
23. Lin, S., Computer solutions of the traveling salesman problem. *Bell System Tech. J.*, 1965, **44,** 2245–2269.
24. Mittenthal, J., Raghavachari, M. and Rana, A. I., A hybrid simulated annealing approach for single machine scheduling problems with non-regular penalty functions. *Computers and Operations Research*, 1993, **20,** 103–111.
25. Morton, T. E. and Pentico, D. W., *Heuristic Scheduling Systems with Applications to Production Systems and Project Management*, John Wiley and Sons, 1993.
26. Ogbu, F. A. and Smith, D. K., The application of the simulated annealing algorithm to the solution of the $n/m/C_{\max}$ flowshop problem. *Computers and Operations Research*, 1990, **17,** 243–253.
27. Ow, P. S. and Morton, T., The single machine early/tardy problem. *Management Science*, 1989, **35,** 177–191.
28. Quaddus, M. A., A generalized model of optimal due date assignment by linear programming. *Journal of the Operational Research Society*, 1987, **38,** 353–359.
29. Reeves, C. (ed.), *Modern Heuristic Techniques for Combinatorial Problems*, Blackwell Scientific Publications, 1993.
30. Sidney, J., Optimal single machine scheduling with earliness and tardiness penalties. *Operations Research*, 1977, **25,** 62–69.
31. Sousa, J. P. and Wolsey, L. A., A time indexed formulation of non-preemptive single machine scheduling problems. *Mathematical Programming*, 1992, **54,** 353–367.
32. Van Laarhoven, P., Aarts, E. and Lenstra, J., Job scheduling by simulated annealing. *Operations Research*, 1992, **40,** 113–125.