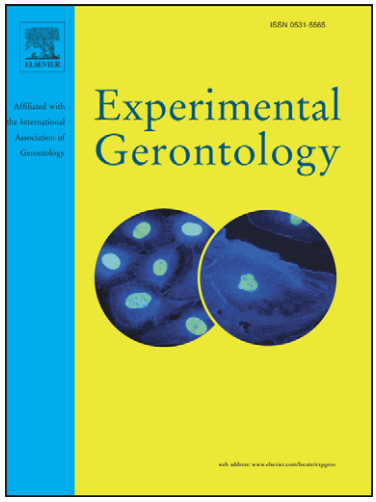
Uric acid as a risk factor for progression of non-diabetic chronic kidney disease? The Mild to Moderate Kidney Disease (MMKD) Study

Gisela Sturm, Barbara Kollerits, Ulrich Neyer, Eberhard Ritz, Florian Kronenberg



Minimizing earliness-tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm

Kuo-Ching Ying

PII: S0531-5565(08)00039-9
DOI: 10.1016/j.exger.2008.01.006
Reference: EXG 8439

To appear in: *Experimental Gerontology*

Received Date: 13 November 2007
Revised Date: 8 January 2008
Accepted Date: 15 January 2008

Please cite this article as: Ying, K-C., Minimizing earliness-tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm, *Experimental Gerontology* (2008), doi: 10.1016/j.exger.2008.01.006

Minimizing earliness-tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm

Kuo-Ching Ying

Department of Industrial Engineering and Management Information, Huafan University, No. 1, Huafan Road, Taipei, Taiwan, R. O. C.

Minimizing earliness-tardiness penalties for common due date single-machine scheduling problems by a recovering beam search algorithm

Abstract

This study considers the NP-hard problem of scheduling jobs on a single-machine against common due dates with respect to earliness and tardiness penalties. An effective and efficient recovering beam search (RBS) algorithm is proposed to solve this problem. To validate and verify the developed algorithm, computational experiments are conducted on a well-known benchmark problem set, and the results are compared with nine meta-heuristics from the relevant literature. The experimental comparison results reveal that the proposed RBS algorithm is a state-of-the-art approach to the single-machine scheduling problem with earliness and tardiness penalties. In terms of both solution quality and computational effort, this study successfully develops a near-optimal approach that will hopefully encourage practitioners to apply it to real-world problems.

Keywords: Scheduling; single-machine; common due date; earliness-tardiness penalties

Corresponding author. Tel.: +886-2-2663-2102.

E-mail address: kcying@huafan.hfu.edu.tw

1. Introduction

In recent decades, production scheduling requirements for meeting a common due date have been investigated extensively, especially since the just-in-time (JIT) philosophy was widely applied in different industries. In general, common due date problems can be classified into two categories: restrictive and unrestrictive ones. If the optimal value of a common due date has to be determined or is given and has no influence on the optimal sequence of jobs, it is called unrestrictive. On the other hand, if a common due date is given and may influence the optimal sequence of the jobs, it is called restrictive. Therefore, a search for an optimal sequence of the jobs has to be carried out with respect to the due date (Feldmann & Biskup, 2003).

Meeting a restrictive common due date is a general scheduling issue in practice. Such instances might occur if a customer orders different variations of a product, all of which must be delivered in the same truck or container ship for transportation cost saving (Sule, 1997), or if a firm has installed a weekly bulk delivery to the wholesaler (Feldmann & Biskup, 2003). A survey of US manufacturing practitioners reveals that meeting a restrictive common due date is one of the most important scheduling criteria (Wisner & Siferd, 1995). When scheduling a production system, jobs completed before the common due date incur earliness penalties, which may include holding costs for finished goods, deterioration of perishable goods and opportunity cost. On the contrary, jobs completed after the common due date sustain tardiness penalties, which may include loss of future sales and rush shipping cost. Besides, the importance of a customer can also be taken into account by using different weights for orders (Dauzère-Pérès & Sevaux, 2004). Hence, in connection with JIT production and delivery, earliness and tardiness penalties are of continuing interest for scheduling researchers and practitioners. However, scheduling problems with a restrictive common

due date where the objective is to minimize earliness-tardiness penalties is not a trivial issue. Even a relatively simple problem, such as the class of a single-machine, is NP-hard (Hall et al., 1991).

This study focuses on restrictive common due date single-machine scheduling problems with respect to total weighted earliness-tardiness penalties. In the standard three-field notation this problem is denoted as $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$. Due to the nature of the complexity of this problem, global optimal solutions cannot be easily obtained when the problem size is large. Thus, for the $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$ problem, no efficient exact methods are known to exist when no restrictions are placed on the values of the penalty per unit time of earliness (α_i) and the penalty per unit time of tardiness (β_i).

Despite this discouraging theoretical result, the past three decades have witnessed growing interest in the $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$ problem. A comprehensive survey on relevant studies of this subject was provided by Gordon et al. (2002). Researchers developed polynomially or pseudo-polynomially solvable algorithms for special cases where conditions for α_i and β_i are imposed (Cheng & Gupta, 1989; Hall et al., 1991; Hoogeveen & van de Velde, 1991; Kahlbacher, 1993). Studies employing meta-heuristics (Lee & Kim, 1995; James, 1997; Feldmann & Biskup, 2003; Hino et al., 2005) and dealing with this argument are even more recent. These recent published meta-heuristics do provide excellent results, but in some cases these meta-heuristics are so intricate that independent coding is unlikely to lead to the same effectiveness or efficiency.

The recovering beam search (RBS) algorithm, first proposed by Della Croce and T'kindt (2002), is one of the most recent and hopeful hybrid heuristics for combinatorial

optimization problems. The RBS algorithm is an enhancement method of beam search (BS), which is a truncated branch and bound algorithm for solving combinatorial optimization problems. In the BS search procedure, only the w most promising nodes at each level of the search tree are retained for further branching, instead of all nodes, and w is the so-called beam width. Meanwhile, the remaining nodes are pruned off permanently. Since only w nodes are retained at each level of the search tree, the running time of BS is polynomial in terms of the problem size (Valente & Alves, 2005).

However, if a node leading to the optimal solution is discarded during the search process, there is no way to later reach the optimal solution using the classic BS. The RBS algorithm seeks to improve upon previous possibly incorrect decisions through the use of a recovering phase that searches for improved partial solutions dominating those selected by the beam. Notably, to guarantee that the total number of explored nodes is polynomial, a partial solution may be only substituted by another partial solution with the same depth level of the search tree in the recovering phase (Della Croce & T'kindt, 2002).

A number of successful applications of the RBS method on scheduling problems (Della Croce et al., 2004; Ghirardi & Potts, 2005; Valente & Alves, 2005; Esteve et al., 2006) have recently appeared in the literature. In an attempt to assist in reducing the gap between theory and practice, this study proposes an RBS algorithm for the $1/d_i = d^{res}/\sum(\alpha_i E_i + \beta_i T_i)$ problem. The remainder of this paper is organized as follows. In the next section, the problem is formulated. Section 3 gives a detailed description of the proposed RBS algorithm. The computational results of applying the proposed RBS algorithm to a famous benchmark problem set are provided in section 4, as well as comparisons of the performances with nine meta-heuristics from the relevant literature. Finally, this study concludes with suggestions for possible future research.

2. Problem formulation

The $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$ problem can be defined as follows. Consider a set of n jobs, each of which requires exactly one operation to be scheduled on a single-machine with a restrictive common due date d. For each job i, the processing time p_i, the penalty per unit time of earliness α_i, and the penalty per unit time of tardiness β_i are given in advance.

A penalty of $\alpha_i E_i$ applies when job i is completed E_i time units earlier than d; whereas a penalty of $\beta_i T_i$ is incurred when it is completed T_i time units later than d. Let C_i be the completion time of job i. The E_i and the T_i for each job i can be calculated by $\max\{d - C_i,\ 0\}$ and $\max\{C_i - d,\ 0\}$, respectively. The objective is to minimize the total weighted earliness-tardiness penalty: $\sum_{i=1}^{n}(\alpha_i E_i + \beta_i T_i)$. Besides these parameters, the following assumptions are made:

- Every job is processed without preemption.
- The setup times of the jobs are included in the processing times and are sequence-independent.
- The ready time of each job is zero; namely all the jobs are available at the beginning of the scheduling period.
- The machine can process no more than one job at a time and is continuously available.

3. The proposed RBS algorithm for the $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$ problem

For a common due date case with general earliness-tardiness penalties, there is an optimal solution satisfying the following three optimality properties. To obtain the

objective function value more efficiently, the following section develops and elaborates on the proposed RBS algorithm by using the three well-established properties.

Property 1. *An optimal schedule does not contain any idle time between any consecutive jobs* (Kahlbacher, 1993).

Property 2. *The optimal schedule is V-shaped around the common due date. In the optimal schedule, the jobs completed before or on the common due date* d *are scheduled in a non-increasing order of the ratios* p_i/α_i, *and the jobs starting on or after* d *are scheduled in a non-decreasing order of the ratios* p_i/β_i (Smith, 1956).

Property 3. *In the optimal schedule, either the first job starts from time zero or the completion time of one job coincides with the common due date* d (Hoogeveen & van de Velde, 1991).

Consider a $1/d_i = d^{res}/\sum(\alpha_i E_i + \beta_i T_i)$ problem with depth size n (i.e. n jobs) of the search tree. A flowchart of the RBS algorithm is depicted in Figure 1. The following subsection will discuss the related steps in detail.

|→ Insert Figure 1 about here ←|

Step 1: Initialization

Set $l = 0$ and $S = \{\sigma_1\}$, where l denotes the search tree level, S represents the vector of the best current partial solutions at a given level, and σ_1 be the parent (root) node.

Step 2: Main loop

A. For each best current partial solution σ_k in S, branch σ_k to generate the corresponding child nodes.

B. Filter procedure:

(a) Add to C the φ child nodes that are selected by the SEA priority evaluation function, where C denotes a set of filtered offspring nodes and φ is a filter width parameter defined by experimental testing.

(b) For each child node in C, compute its evaluation function value $V=(1-\gamma)LB+\gamma UB$, where γ $(0\le\gamma\le1)$ denotes the upper bound weight in the weighted sum of a lower bound (LB) and an upper bound (UB) on the optimal solution value of that partial solution.

(c) Sort the child nodes in C in a non-decreasing order of their evaluation function values and add to T the w child nodes with the best evaluation function values, where T denotes a set of beam nodes and w is a beam width parameter defined by experimental testing.

C. Recovering phase:

(a) For each partial solution represented by the node in T, pick the last job in the job list and find the best sequence by placing that job in all possible positions in the partial solution.

(b) Update the vector of the best current partial solutions S by the w partial solution found so far.

Step 3: Stopping criterion:

Let $l=l+1$, IF $l<n$: GO TO step 2; ELSE STOP: rearrange each partial solution in S according to properties 1~3 and find the best solution.

3.1. The SEA priority evaluation function

Recently, we proposed a quick method, namely the sequential exchange approach (SEA; Lin et al., 2007), to obtain a near optimum solution for $1/d_i = d^{res}/\sum(\alpha_i E_i + \beta_i T_i)$ problems. This simple heuristic firstly divides the jobs into two sets: S_E and S_L, where jobs in set S_E start processing before the common due date, and jobs in set S_L start processing on or after the common due date. Then, SEA swaps or moves jobs between the two sets systematically to derive a near optimum solution.

The priority evaluation function of the filter procedure at each search tree level is a key issue in the RBS technique. In this study, a computationally inexpensive filtering procedure, namely the SEA priority evaluation function is applied to select some of the children of each beam node for a more detailed evaluation. Assume that a near optimum solution π is obtained by the SEA. Based on the job j's order number ($[j]$) in S_E or S_L of solution π, we set the priority evaluation function value of job (node) j ($PEFV_j$) as follows:

$$PEFV_j = \begin{cases} n-[j]+1, & \text{if job } j \in S_E \\ [j], & \text{if job } j \in S_L \end{cases}$$

Then, the φ filtered child nodes with the best SEA priority evaluation function values are added to set C for further evaluation.

3.2. The evaluation function

After being filtered by the SEA priority evaluation function, the φ selected child nodes are then accurately evaluated via a function V defined by $V = (1-\gamma)LB + \gamma UB$, where γ is a given weight and LB and UB refer to a lower bound and an upper bound, respectively.

In this study, we propose the following heuristic to obtain UB of the partial schedule π' represented by the node. The idea for this heuristic is based on work completed by Dileepan (1993) and Biskup and Feldmann (2001).

Heuristic of *UB*. Let S_A and S_B be the ordered sets of jobs that are started after the common due date and completed before the common due date, respectively. The main steps of the UB heuristic are as follows.

Step 1: Sequentially assign the jobs of the partial schedule π' into S_B until the time gap $T_{gap} = d - \sum_{i \in S_B} p_i$ is smaller than the assigned job's processing time. Then, add the remaining jobs of π' to set S_A in a non-decreasing order of p_i / β_i.

Step 2: IF $S_A = \varnothing$: GO TO step 3; ELSE: GO TO step 5.

Step 3: Add to set $P^{\alpha\beta}$ all the jobs which are not included in π' according to the decreasing ratios β_i / α_i. The tiebreak rule gives preference to the job with the smallest value of β_i.

Step 4: Iteratively assign the first job of $P^{\alpha\beta}$ to set S_B and delete it from $P^{\alpha\beta}$ until one of the following situations occurs:

A. The time gap T_{gap} is smaller than the assigned job's processing time. If in this case $T_{gap} > 0$, sequentially search for another job of $P^{\alpha\beta}$ which fits into the gap. As soon as $T_{gap} = 0$ or no more jobs of $P^{\alpha\beta}$ fit into the gap, assign the remaining jobs of $P^{\alpha\beta}$ to set S_A. Finally, sequence all jobs in S_A and S_B according to property 2 (the V-shaped property).

B. $n/2$ jobs have been assigned to set S_B. In this case iteratively assign the first of the remaining jobs of $P^{\alpha\beta}$ to set S_B until the objective function value is increased. Terminate the heuristic and the current objective function value is saved as UB.

Step 5: Shift the jobs in S_B and S_A to make the completion time of the final job in S_B and the start time of the initial job in S_A equal to the common due date d. Terminate the heuristic and the current objective function value is saved as UB.

To get LB of the evaluation function V, we formulate the $1/d_i = d^{res}/\sum(\alpha_i E_i + \beta_i T_i)$ problem, and decompose the problem into a subproblem based on the Lagrangian relaxation, and further develop an efficient multiplier adjustment method to compute the values of the Lagrangian multipliers. The $1/d_i = d^{res}/\sum(\alpha_i E_i + \beta_i T_i)$ problem can be logically formulated as an integer programming problem (**P**):

$$Z = \min \sum_{i=1}^{n} (\alpha_i E_i + \beta_i T_i)$$

$$\text{s.t.} \quad T_i = s_i + p_i - d + E_i, \quad i = 1, 2, ..., n, \tag{1}$$

$$s_i + p_i \le s_k + M(1 - x_{ik}), \quad i = 1, 2, ..., n-1; \; k = i+1, ..., n, \tag{2}$$

$$s_k + p_k \le s_i + M x_{ik}, \quad i = 1, 2, ..., n-1; \; k = i+1, ..., n, \tag{3}$$

$$T_i, E_i, s_i \ge 0, \quad i = 1, 2, ..., n, \tag{4}$$

$$x_{ik} \in \{0, 1\}, \quad i = 1, 2, ..., n-1; \quad k = i+1, ..., n. \tag{5}$$

The values for earliness and tardiness are calculated by constraint (1). Constraints (2) and (3) determine the starting times of the jobs. If job i is sequenced prior to job

k the constraint $s_i + p_i \le s_k$ only holds if $x_{ik} = 1$. Due to the addition of big number M, constraint (3) is not restrictive with $x_{ik} = 1$. Alternatively, for $x_{ik} = 0$, constraint (3) gives $s_k + p_k \le s_i$ and constraint (2) is not restrictive.

A Lagrangian relaxation of constraint (1) yields the Lagrangian subproblem (**LR**):

$$\textbf{(LR)} \quad L(\lambda) = \min \sum_{i=1}^{n} [\lambda_i (s_i + p_i - d) + (\alpha_i + \lambda_i) E_i + (\beta_i - \lambda_i) T_i]$$
$$\text{s.t.} \quad (2) - (5),$$

where $\lambda = (\lambda_1, \lambda_2, ..., \lambda_n)$ are the Lagrangian relaxation multipliers.

If $\alpha_i + \lambda_i < 0$ or $\beta_i - \lambda_i < 0$ for some i ($i = 1, 2, ..., n$), then we have $E_i = \infty$ or $T_i = \infty$, giving $L(\lambda) = -\infty$. To avoid this, we therefore require that $-\alpha_i \le \lambda_i \le \beta_i$ for each i ($i = 1, 2, ..., n$). The Lagrangian subproblem (**LR**) is solved by setting $E_i = T_i = 0$ for each i, where $i = 1, 2, ..., n$, and then scheduling the jobs in a non-increasing order of λ_i / p_i according to Smith's rule (Smith, 1956).

In general, the Lagrangian multipliers can be iteratively updated using the subgradient optimization method (SOM) to improve the lower bound. However, experiments have shown that it takes a lot of time to get the SOM to converge (Sourd & Kedad-Sidhoum, 2003). The speed increases of the multiplier adjustment method (MAM) more than sufficiently compensate for the possible loss in lower quality over the SOM, we resort to using the MAM.

The MAM first requires a heuristic to sequence the jobs, and then chooses the multipliers so that the resulting lower bound is as large as possible. Let s_i^* be the start time of job i ($i = 1, ..., n$) in the job sequence generated by the heuristic. To obtain the maximum value of **LR**, we solve the problem **LR**(s_i^*):

$$W = \max \sum_{i=1}^{n} \lambda_i (s_i^* + p_i - d)$$

(LR(s_i^*)) s.t. $\lambda_i / p_i \geq \lambda_{i+1} / p_{i+1}$, $i \in$ early jobs, (6)

$\lambda_i / p_i \leq \lambda_{i+1} / p_{i+1}$, $i \in$ tardy jobs, (7)

$-\alpha_i^* \leq \lambda_i \leq \beta_i^*$, $i = 1, 2, ..., n.$ (8)

where $-\alpha_i^* = p_i \max\{-\alpha_k / p_k, k = i, i+1, ..., n\}$ and $\beta_i^* = p_i \min\{\beta_k / p_k, k = 1, 2, ..., i\}$.

To obtain the Lagrangian multipliers more rapidly, we use the weighted shortest processing time (WSPT) and the weighted longest processing time (WLPT) to generate the initial job sequence and s_i^*. Then, we calculate $R_i = s_i^* + p_i - d$ ($i = 1, 2, ..., n$), and set the Lagrangian multipliers as follows:

$$\lambda_i = \beta_i^*, \quad \text{if } R_i \geq 0,$$

$$\lambda_i = -\alpha_i^*, \quad \text{if } R_i < 0.$$

After obtaining the Lagrangian multipliers, we can get the solution to the problem **LR(s_i^*)** and set it as the LB of the evaluation function V.

4. Computational results and discussion

4.1. Test problems

One difficulty faced by researchers in scheduling is to compare their developed heuristics with those of other researchers. If a standard set of test problems is accessible, different algorithms' performances can be compared on exactly the same set of test problems. For this reason we chose 280 benchmark problems from Biskup and Feldmann (2001) as the test problems for this study.

Biskup and Feldmann have generated benchmark problems with the number of jobs $n = 10, 20, 50, 100, 200, 500$ and 1000 for the $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$

problem. The restrictive factor h for determining the common due date d, which can be computed as $d = \lfloor h \sum p_i \rfloor$, was set to be 0.2, 0.4, 0.6 and 0.8. For each combination of n and h, ten instances were generated, respectively. Thus, the test bed comprised 280 problems.

4.2. Computational results

The choice of parameters may affect the quality of the results. Choosing appropriate values for the parameters is time consuming and in general seems dependent on the set of data. A number of preliminary experiments were performed to fine tune the parameters of the proposed RBS algorithm. The experiments conducted here reveal that the filter width φ and the beam width w yielded better quality within reasonable computational expenses at values of around 6 and 3, respectively. The experiments also indicate that the parameter γ yielded better quality at values of around 0.3, 0.4, 0.5, and 0.6 for $h = 0.2$, 0.4, 0.6 and 0.8 problems, respectively. Consequently, these values were used for the experiments conducted in this study.

The proposed RBS algorithm was coded in Visual C#.net, and run on a PC with an Intel Pentium 4 (2.4GHz) CPU. The final results of the proposed RBS algorithm compared with nine meta-heuristics proposed by Feldmann and Biskup (2003) and Hino et al. (2005) are shown in Table 1. Feldmann and Biskup investigated the same benchmark problems by applying five different meta-heuristics, namely evolutionary strategy (ES), ES with a destabilization phase (ESD), simulated annealing (SA), threshold accepting (TA) and TA with a back step (TAR). However, they only considered the problems with the restrictive factor h set as 0.2 and 0.4. Each meta-heuristics was performed three runs on each single instance with different initial random numbers. The data listed in Table 1 are the best objective function values

obtained from totaling 15 runs of the five different meta-heuristics (FDM).

Hino et al. (2005) proposed four meta-heuristics, including tabu search (TS), genetic algorithm (GA), hybrid of tabu search and genetic algorithm (HTG), and hybrid of genetic algorithm and tabu search (HGT) to the same benchmark problems. Each instance was solved over ten trials by each of these four meta-heuristics and the best solutions obtained were recorded.

|→ Insert Table 1 about here ←|

To compare the effectiveness of the different approaches, the solution quality is measured by listing the average improvement rate (%) of the objective function values from the upper bounds proposed by Biskup and Feldmann (2001). As revealed in Table 1, on the whole, the proposed RBS algorithm outperformed the nine meta-heuristics. Total average improvement rates of 6.72 and 4.37 for the problems with the restrictive factors $h = 0.2,\ 0.4,$ and $h = 0.2,\ 0.4,\ 0.6,\ 0.8$ were achieved, respectively. The proposed RBS algorithm was superior in all of the different sized problems with different values for the restrictive factor. Notably, for all of the $n = 10$ instances, the optimal solutions (determined by applying LINGO to their integer programming models) were obtained by the proposed RBS heuristic, too. These analytical results clearly indicate that the proposed algorithm is a promising approach for solving the $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$ problem.

The average computation time (CPU time in seconds) of the proposed RBS algorithm run on a Pentium 4 (2.4GHz) PC and that of the five different meta-heuristics (FDM) proposed by Feldmann and Biskup (2003) run on a Pentium 90 PC are presented in Table 2. It should be noted that the FDM proposed by Feldmann and Biskup would in general require multiple trials with different random seeds to get a better solution.

|→ Insert Table 2 about here ←|

As revealed in Table 2, the computation time required by the proposed algorithm is very short. Since the computation times vary with hardware, software and coding, this study did not directly compare computational efficiency. Nonetheless, even for problems involving up to 1000 jobs, the proposed RBS algorithm can obtain very good solutions within reasonable computational expenses. This indicates that the proposed RBS heuristic can be applied to real world problems.

On the other hand, as the running time of the meta-heuristics proposed by Hino et al. (2005) is provided as a graph, therefore, only approximate values of the GA and the hybrid meta-heuristics (i.e. HTG and HGT) can be used for comparison. The average running time of $n = 1000$ is over 90 and 32 seconds of the GA and the hybrid meta-heuristics on a Pentium 4 (1.7GHz) PC, respectively. Notably, the best objective function values of these meta-heuristics were obtained from ten runs while the objective function values of the proposed RBS algorithm were obtained from one trial. These results indicated that the proposed RBS algorithm is a more efficient and effective approach compared with these state-of-the-art meta-heuristics.

5. Conclusions

Single-machine is one of the most useful models in practice, since a complex system can be reduced to a single-machine problem, especially if there is a bottleneck machine in the system. The purpose of this study is to make a step towards establishing an effective heuristic for the $1/d_i = d^{res} / \sum(\alpha_i E_i + \beta_i T_i)$ problem. In this study, we propose an RBS algorithm for this problem. The experimental results clearly indicate that the proposed RBS algorithm is a state-of-the-art method for solving this problem,

as seen by comparing the obtained results to the best available meta-heuristics on a wide range of benchmark instances. Further, the experimental results also indicate that the proposed approach can be applied to efficiently schedule common due-date problems with general earliness-tardiness penalties of a realistic size.

Possible directions for future research include hybridizing the RBS algorithm with meta-heuristics, such as simulated annealing (SA), genetic algorithm (GA), tabu search (TS), and ant colony optimization (ACO), and also by extending the proposed RBS algorithm to assist in solving multi-machine scheduling problems with general or non-linear earliness-tardiness penalties in regards to a common due date.

Acknowledgements

The authors would like to thank Biskup and Feldmann [19] for their benchmark problem sets and solution files. This paper was financial support in part by the National Science Council of the Republic of China, Taiwan, under the Contract No. NSC 96-2221-E-211-009.

References

Biskup, D., & Feldmann, M. (2001). Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates. *Computers & Operations Research*, *28*, 787-801.

Cheng, T. C. E., & Gupta, M. C. (1989). Survey of scheduling research involving due date determination decisions. *European Journal of Operational and Research*, *38*, 156-166.

Dauzère-Pérès, S., & Sevaux, M. (2004). An exact method to minimize the number of tardy jobs in single machine scheduling. *Journal of Scheduling*, *7*, 405-420.

Della Croce, F., & T'kindt, V. (2002). A Recovering Beam Search algorithm for the one-machine dynamic total completion time scheduling problem. *Journal of Operational Research Society*, *53*, 1275-1280.

Della Croce, F., Ghirardi, M., & Tadei, R. (2004). Recovering Beam Search: Enhancing the beam search approach for combinatorial optimization problems. *Journal of Heuristics*, *10*, 89-104.

Dileepan, P. (1993). Common due date scheduling problem with separate earliness and tardiness penalties. *Computers & Operations Research*, *20*, 179-184.

Esteve, B., Aubijoux, C., Chartier, A., & T'kindt, V. (2006). A recovering beam search algorithm for the single machine Just-In-Time scheduling problem. *European Journal of Operational Research*, *172*, 798-813.

Feldmann, M., & Biskup, D. (2003). Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches. *Computers & Industrial Engineering*, *44*, 307-323.

Ghirardi, M., & Potts, C. N. (2005). Makespan minimization for scheduling unrelated parallel machines: A recovering beam search approach. *European Journal of Operational Research*, *165*, 457-467.

Gordon, V., Proth, J. M., & Chu, C. (2002). A survey of the state-of-art of common due date assignment and scheduling research. *European Journal of Operational Research*, *139*, 1-25.

Hall, N. G., Kubiak, W., & Sethi, S. P. (1991). Earliness-tardiness scheduling problems

II: Deviation of completion times about a restrictive common due date. *Operations Research*, *39*, 847-856.

Hino, C. M., Ronconi, D. P., & Mendes, A. B. (2005). Minimizing earliness and tardiness penalties in a single-machine problem with a common due date. *European Journal of Operational Research*, *160*, 190-201.

Hoogeveen, J. A., & van de Velde, S. L. (1991). Scheduling around a small common due date. *European Journal of Operational Research*, *55*, 237-242.

James, R. J. W. (1997). Using tabu search to solve the common due date early/tardy machine scheduling problem. *Computers & Operations Research*, *24*, 199-208.

Kahlbacher, H. G. (1993). Scheduling with monotonous earliness and tardiness penalties. *European Journal of Operational Research*, *64*, 258-277.

Lee, C. Y., & Kim, S. J. (1995). Parallel genetic algorithms for the earliness-tardiness job scheduling problem with general penalty weights. *Computers & Industrial Engineering*, *28*, 231-243.

Lin, S.W., Chou, S. Y., & Ying, K. C. (2007). A sequential exchange approach for minimizing earliness-tardiness penalties of single-machine scheduling with a common due date. *European Journal of Operational Research*, *177*, 1294-1301.

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, *3*, 59-66.

Sourd, F., & Kedad-Sidhoum, S. (2003). The one-machine problem with earliness and tardiness penalties. *Journal of Scheduling*, *6*, 533-549.

Sule, D. R. (1997). *Industrial scheduling*. PWS publishing company, Boston, MA.

Valente, J. M. S., & Alves, R. A. F. S. (2005). Filtered and recovering beam search algorithms for the early/tardy scheduling problem with no idle time. *Computers & Industrial Engineering*, *48*, 363-375.

Wisner, J. D., & Siferd, S. D. (1995). A survey of US manufacturing practices in make-to-order machine shops. *Production and Inventory Management*, *1*, 1-7.

Figure 1. Flowchart of the RBS algorithm

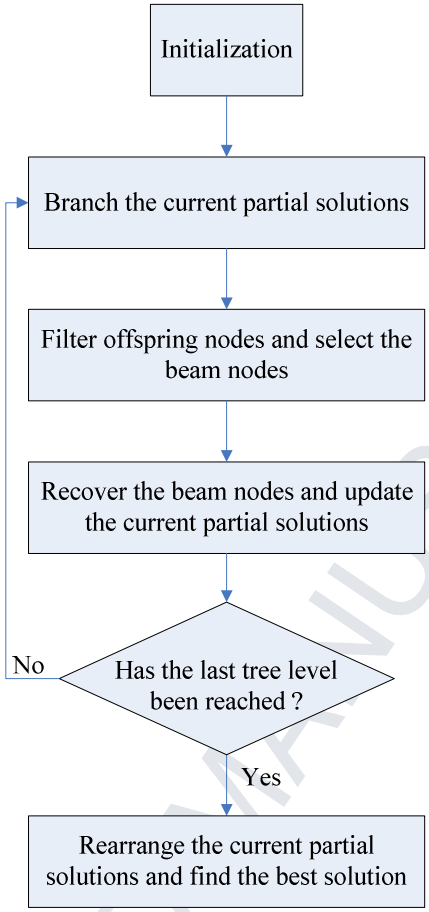


Table 1. Average improvement rate (%) of meta-heuristics and the proposed RBS algorithm from upper bounds

n	h	FDM	TS	GA	HTG	HGT	RBS
10	0.2	**6.19**	5.95	6.07	6.07	6.07	**6.19**
	0.4	**2.17**	1.93	1.98	1.98	1.98	**2.17**
	0.6	—	-0.08	0.00	0.00	0.00	**0.02**
	0.8	—	0.15	**0.15**	**0.15**	**0.15**	**0.15**
20	0.2	3.84	3.84	3.84	3.84	3.84	**6.53**
	0.4	1.63	1.62	1.62	1.62	1.62	**4.31**
	0.6	—	0.71	0.68	0.71	0.71	**2.31**
	0.8	—	0.41	0.28	0.41	0.41	**2.76**
50	0.2	5.65	5.70	5.68	5.70	5.70	**7.25**
	0.4	4.64	4.66	4.60	4.66	4.66	**6.35**
	0.6	—	0.32	0.31	0.27	0.31	**3.93**
	0.8	—	0.24	0.19	0.23	0.23	**2.54**
100	0.2	6.18	6.19	6.17	6.19	6.19	**8.97**
	0.4	4.94	4.93	4.91	4.93	4.93	**6.44**
	0.6	—	0.01	0.12	-0.08	-0.04	**1.98**
	0.8	—	0.15	0.12	0.08	0.11	**2.33**
200	0.2	5.73	5.76	5.74	5.76	5.76	**7.76**
	0.4	3.79	3.74	3.75	3.75	3.75	**5.98**
	0.6	—	0.01	0.13	-0.37	-0.07	**2.87**
	0.8	—	0.04	0.14	-0.26	-0.07	**1.78**
500	0.2	6.40	6.41	6.41	6.41	6.41	**8.82**
	0.4	3.52	3.57	3.58	3.58	3.58	**7.11**
	0.6	—	-0.25	0.11	-0.73	-0.15	**2.35**
	0.8	—	-0.21	0.11	-0.73	-0.13	**1.58**
1000	0.2	6.72	6.73	6.75	6.74	6.74	**8.95**
	0.4	4.30	4.39	4.40	4.39	4.39	**7.27**
	0.6	—	-1.01	0.05	-1.28	-0.42	**1.11**
	0.8	—	-1.13	0.05	-1.28	-0.40	**2.65**
Total average for h = 0.2 and 0.4		4.69	4.67	4.68	4.69	4.69	**6.72**
Total average for h = 0.2, 0.4, 0.6 and 0.8		—	2.31	2.43	2.24	2.37	**4.37**

Table 2. The average computation time of the proposed RBS algorithm and the five different meta-heuristics (FDM) proposed by Feldmann and Biskup

	$n = 10$	$n = 20$	$n = 50$	$n = 100$	$n = 200$	$n = 500$	$n = 1{,}000$
RBS: $h = 0.2$	< 0.01	< 0.01	0.021	0.15	1.12	15.37	232.10
RBS: $h = 0.4$	< 0.01	< 0.01	0.025	0.14	1.11	15.46	228.79
RBS: $h = 0.6$	< 0.01	< 0.01	0.024	0.14	1.12	15.66	230.44
RBS: $h = 0.8$	< 0.01	< 0.01	0.028	0.15	1.11	15.48	229.56
Average of RBS	< 0.01	< 0.01	0.024	0.145	1.115	15.49	230.22
Average of FDM	0.9	47.8	87.3	284.9	955.2	3,547.2	10,962.5