Priority rule-based reconstruction for total weighted tardiness minimization of job-shop scheduling problem

Soichiro YOKOYAMA*, Hiroyuki IIZUKA* and Masahito YAMAMOTO*

*Graduate School of Information Science and Technology, Hokkaido University
14 Kita, 9 Nishi, Kita-ku, Sapporo, Hokkaido 060-0814, Japan
E-mail: yokoyama@complex.ist.hokudai.ac.jp
Received 24 February 2014

Abstract
We propose an efficient heuristic method for job-shop scheduling problems (JSP) with the objective of total weighted tardiness minimization. The proposed method uses schedule reconstructions by priority rules to guide a local search towards promising solutions. Typically priority rules determine the whole schedule and thus incorporating it with local search procedure is difficult. In our proposed method, a priority rule decides a schedule within an arbitrary selected time window and the rest of the schedule is determined by the schedule obtained by a conventional local search method. The priority rule is given by a linear combination of simple priority rules. To improve a schedule efficiently, an appropriate set of time window and gains of the linear combined priority rule is required. Therefore, we optimize the set of time window and rule gains with genetic algorithm. This rule-based reconstruction procedure and the conventional local search procedure are alternately applied to a current solution. In the experiments, the efficiency of rule-based reconstruction procedure is verified and the proposed method is compared with one of the most effective existing methods. The results show that the proposed method outperforms the existing method on large problems with sufficient computational time. The average performance is particularly improved due to the ability of rule-based reconstruction procedure to escape from local optima of the conventional local search procedure efficiently.

Key words: Job-shop scheduling problem, Total weighted tardiness minimization, Local search, Priority rule, Genetic algorithm

1. Introduction

The job-shop scheduling problem (JSP) is one of the representative combinatorial optimization problems where the objective is to find an ideal assignment of resources when multiple jobs must be processed on multiple machines simultaneously. JSP has many application fields such as production planning and logistics. Thus, JSP has attracted many researchers’ attention. Most studies consider the minimization of the maximum completion time of all jobs or makespan criterion. Genetic algorithm (Bierwirth, 1995) and tabu search algorithm (Taillard, 1994) perform well on such a problem. Further improvements are obtained by combining genetic algorithm and local search (Gao, et al., 2011) or refining neighborhood structure (Nowicki and Smutnicki, 1996) (C. Zhang, et al., 2007). In practice, due date related performance is often more important and thus total tardiness minimization is also actively studied.

In this paper, the minimization of total weighted tardiness in the job-shop environment is considered. Due dates and weights are assigned to each job and tardiness penalty is given with respect to the weight if the completion time is later than the due date. There are roughly two kinds of approaches for the minimization of total weighted tardiness. First one is using iterated local search procedures proposed by Kreipl (2000) and Essafi et al. (2008), which are quite efficient in terms of performance on standard benchmark problems (Pinedo and Singer, 1999). The large step random walk proposed by Kreipl (2000) consists of iterative improvement to intensify search and random walk to escape from the local optima. The genetic local search (GLS) proposed by Essafi et al. (2008) optimizes initial solutions for the iterative local search by genetic algorithm and is considered to be one of the most effective methods. They are also superior to the shifting bottleneck heuristic (Pinedo and Singer, 1999) which decomposes JSP into multiple sub-problems of determining...
assignments on a single machine and are considered to be ones of the most efficient approaches to JSP with total weighted tardiness criterion.

Another approach is based on priority rules that have been extensively studied especially for tardiness objective. Early works are reviewed by Panwalkar and Iskander (1977). These priority rules determine the sequences of operations on machines according to various information about operations such as processing time, remaining operation number and due date. The goal of the studies is to develop an efficient priority rule to obtain better solutions in different problems and Vepsalainen and Morton (1987) confirmed that appropriate combination of processing time and due date leads to a priority rule which performs better than other priority rules on average. However, priority rules are generally less efficient compared to local search algorithms because the operation sequences are optimized for a particular problem instance in local search algorithms. Recently, Zhou et al. (2009) and R. Zhang et al. (2013) exploit priority rules to guide the search for promising schedule for a particular instance. Zhou et al. (2009) determine the first operations of the sequences by genetic algorithm while the rest of the sequences are determined with a priority rule. Thus, the search space of genetic algorithm is reduced by the priority rule to improve its computational efficiency. R. Zhang et al. (2013) extract scheduling policies from a feasible schedule in a form of priority rules and feasible schedules are generated from the policies to obtain a more efficient schedule. However, their performance is worse than state-of-the-art local search procedures on small benchmark problems and their solution representation is so different from the one used in local search algorithms that their methods have not been incorporated with efficient local search algorithms.

We propose an efficient heuristic method for the minimization of total weighted tardiness by incorporating priority rule with local search methods. Conventionally, the whole schedule are determined with a priority rule. Therefore, priority rules have been used only to generate initial solutions of local search methods. In this paper, rule-based reconstruction procedure within arbitrary selected time window is proposed to modify a feasible schedule with priority rules and to generate neighborhood schedules with priority rules during local search procedure. This rule-based reconstruction procedure defines a sub-problem for a feasible schedule and provides another feasible schedule by solving it with priority rules. The sub-problem is defined by a decision of machine assignments of jobs within an arbitrary selected time window where other assignments are determined according to the feasible schedule. Since Tamura et al. (2013) shows that linearly-combined priority rules are more effective for makespan criterion if the gains are optimized for particular instance, the linear-combined priority rules are adopted in our rule-based reconstruction procedure. The time window and the gains of linear-combines priority rules are optimized by genetic algorithm because most of the combination provides a schedule with worse tardiness and we are interested only in improving the tardiness of the schedule. In our proposed method, local search procedure in GLS is adopted and rule-base reconstruction procedure above is employed to escape from local optima. As a result, our proposed method guides the conventional local search procedure towards more promising solutions by priority rules and thus our proposed method outperforms GLS which shows best performance so far.

The rest of the paper is organized as follows. JSP with tardiness criterion is formulated in Section 2. Section 3 describes the conventional local search procedure we adopted in our method. In Section 4, our proposed method is explained. Priority rule-based reconstruction procedure is presented and combined with the conventional local search procedure. Computational experiments are performed on benchmark problems to compare the proposed method with existing algorithms and the results are shown in Section 5 and the effectiveness of the proposed rule-based reconstruction method is also investigated. Section 6 concludes the paper.

2. Job-shop Scheduling Problem with weighted tardiness objective

We consider job-shop environment with n jobs $J_1, J_2, \ldots, J_n$ and m machines $M_1, M_2, \ldots, M_m$. A job $J_i$ $(1 \leq i \leq n)$ consists of m operations $o_{i1}, o_{i2}, \ldots, o_{im}$, weight $w_i$ and due date $d_i$. An operation $o_{ij}$ is processed on a machine $r_{ij} \in \{M_1, M_2, \ldots, M_m\}$ for processing time $p_{ij} \geq 0$ without preemption. Each machine can process at most one operation at the same time and thus operations on the same machine must be processed in sequence. Each job must be processed on every machine exactly once. Thus, we assume that $r_{ij} \neq r_{ik}$ $(j \neq k \forall j, k)$. Additionally, operations in the same job must be processed in the predetermined technological order. An operation $o_{ij}$ is $j$-th operation in $J_i$ and preceding operations $o_{i1}, o_{i2}, \ldots, o_{ij-1}$ must be finished before processing $o_{ij}$. An example with three jobs and three machines is shown in Table 1. Weights, due date, technological order and processing time on each machines are given for each job.

A schedule is specified by enumerating starting time $s_{ij}$ for every operation $o_{ij}$ and the schedule is feasible if and only if equations below are satisfied.

$$0 \leq s_{ij} \quad (i = 1, 2, \ldots, n)$$

(1)
Table 1 An instance of JSP with 3 jobs and 3 machines

| Job | Weight | Due date | 1st operation | 2nd operation | 3rd operation |
|-----|--------|----------|---------------|---------------|---------------|
| J1  | 1      | 14       | (M1, 2)       | (M1, 3)       | (M2, 4)       |
| J2  | 2      | 9        | (M1, 3)       | (M2, 2)       | (M6, 4)       |
| J3  | 4      | 12       | (M1, 1)       | (M1, 3)       | (M2, 2)       |

\[
s_{i,j} + p_{i,j} \leq s_{i,j+1} \quad (i = 1, 2, \ldots, n, \ j = 1, 2, \ldots, m - 1)
\]

\[
s_{i,j} + p_{i,j} \leq s_{p,q} \quad \text{or} \quad s_{p,q} + p_{p,q} \leq s_{i,j} \quad (1 \leq i,j \leq n, \ 1 \leq p,q \leq m, \ a_{i,j} \neq a_{p,q}, \ r_{i,j} = r_{p,q})
\]

Equation (1) states that all operations are released on time 0. Equations (2) and (3) ensure that technological order is held for every job and each machine does not process multiple operations at the same time, respectively. Completion time \(C_i\) of job \(J_i\) is given by \(C_i = s_{i,m} + p_{i,m}\) and \(J_i\) is tardy if \(C_i > d_i\). One of feasible schedules for the instance shown in Table 1 is given by \(s_{1,1} = 0, s_{1,2} = 2, s_{1,3} = 7, s_{2,1} = 2, s_{2,2} = 5, s_{2,3} = 7, s_{3,1} = 5, s_{3,2} = 6, s_{3,3} = 11\). Completion time of each job is \(C_1 = 11, C_2 = 11\) and \(C_3 = 13\). \(J_2\) and \(J_3\) are completed after their due dates and thus \(J_2\) and \(J_3\) are tardy jobs. This schedule is shown in Fig. 1 using Gantt chart where occupation of machines by each operation is presented with time as the horizontal axis.

![Gantt chart](image)

The tardiness \(T_j\) for job \(J_j\) is defined by tardiness \(T_j = \max[C_j - d_j, 0]\) with completion time and due date for the job. Thus only tardy jobs whose completion time is later than the due date affect the total weighted tardiness. Weight \(w_i\) represents importance of job \(J_i\), and higher value indicates that the delay of the job is heavily penalized. Overall, the objective function is the minimization of the total weighted tardiness \(TWT\) shown by the following equation.

\[
TWT = \sum_t w_i T_i
\]

Note that in some cases it is impossible to complete all jobs in time depending on the tightness of due date and thus total weighted tardiness of optimal schedules differs by instances. The schedule shown in Fig. 1 has two tardy jobs \(J_2\) and \(J_3\). Their tardiness values are \(T_2 = 11 - 9\) and \(T_3 = 13 - 12\) respectively and their weights are given by \(w_2 = 2\) and \(w_3 = 4\). Therefore, the total weighted tardiness of the schedule is \(2 \times 2 + 4 \times 1 = 8\).

The set of all feasible schedules have three subsets, semi-active schedule, active schedule and non-delay schedule. Semi-active schedules are a set of schedules whose operations cannot be processed at an earlier time without modifying the sequence of the operations on each machine or breaking the constraints of schedule feasibility. The schedule shown in Fig. 1 is included in semi-active schedule. Active schedules are a set of schedules where each operation cannot be processed at an earlier time without delaying other operations or breaking the constraints. The schedule shown in Fig. 1 is excluded from active schedule because operation \(o_{3,1}\) could be processed on \(t = 0\) without delaying \(o_{1,2}\) by exchanging the operation sequence on machine \(M_3\). In Non-delay schedule, no machine is kept idle when it could start processing some operation. The schedule in Fig. 1 is not a non-delay schedule because \(M_3\) is kept idle on \(t = 0\) although \(o_{3,1}\) could be processed. Since active schedule is clearly a subset of semi-active schedule and it is proved that active schedule includes one of the optimal schedules, most methods search for an optimal schedule from a set of active schedule. GT algorithm proposed by Giffler and Thompson (1960) can enumerate any active schedule and thus GT algorithm is employed in methods whose solution space consists of active schedule. In the local search procedure, however, semi-active schedule is considered because the neighborhood solutions consist of semi-active schedule can be obtained much faster. Although non-delay schedule may not contain the optimal schedules, focusing on non-delay schedule can sometimes yields better result if the computational time is restricted. Therefore, we consider semi-active schedule in our local search procedure and focus on active schedule during the priority rule-based reconstruction procedure.
3. Local search procedure

Recent studies show that local search procedures are useful as an efficient heuristic for JSP with both makespan objective and tardiness objective. For the makespan criterion, tabu search algorithms such as Nowicki and Smutnicki (1996) and C. Zhang et al. (2007) are known to be very effective. Genetic algorithm approaches also employ some forms of local search procedures and performance improvements are confirmed regardless of solution representation schemes in (Gao, et al., 2011), (Qing-dao-er-ji and Wang, 2012) and (Hasan, et al., 2009). For the tardiness criterion, all tardy jobs affect the value of the objective function and thus slightly modified neighborhood structure is used. Kreipl (2000) developed a large-step random-walk algorithm which is proven to be very effective for tardiness minimization. Essafi et al. (2008) applies the local search procedure to every chromosome of genetic algorithm and this is one of the best heuristics for JSP with tardiness objective so far.

Therefore, the local search procedure is adopted as a basis of our proposed method. In most of the successful neighborhood structures, a feasible solution is represented by a disjunctive graph and neighborhood is defined by the changes of operation sequences processed on the same machine. In this section, disjunctive graph representation for a tardiness minimization by Pinedo and Singer (1999) is presented. Neighborhoods are defined with the disjunctive graph model and the local search procedure of Essafi et al. (2008) which is used in our proposed method is explained.

3.1. Disjunctive graph representation

A job-shop scheduling problem is represented by a disjunctive graph $G = (V, A, E)$. The vertex set $V$ consists of the set of operations, dummy source nodes $O$ and sink nodes for each job $s_i$ ($i = 1, 2, \ldots, n$). The directed arc set $A$ represents technological order and connects consecutive operation in the same job. The disjunctive graph set $E$ connects every pair of operations processed on the same machine. Directed arcs whose source node is $o_{i,j}$ has a weight of $p_{i,j}$. The problem instance shown in Table 1 is represented by the disjunctive graph in Fig. 2.

![Fig. 2 Disjunctive representation of the instance shown in Table 1](image)

One of semi-active schedules is obtained by determining direction of each disjunctive graph. The direction represents the sequence in which operations are processed on the particular machine. Thus, this is equivalent to determine the processing order of operations on each machine. A corresponding schedule is feasible if and only if the graph is acyclic. Starting time of operation $o_{i,j}$ in the schedule is given by the length of the longest path from the source node $O$ to the associated node of $o_{i,j}$. Therefore, a completion time of job $J_i$ is the length of the longest path from $O$ to a sink node $s_i$. A feasible schedule for the instance in Fig. 2 is shown in Fig. 3(a) and corresponding Gantt chart for the schedule is shown in Fig. 3(b). In this example, processing order on $M_1, M_2, M_3$ is given by $o_{1,1} \rightarrow o_{2,1} \rightarrow o_{3,2}, o_{2,2} \rightarrow o_{3,3} \rightarrow o_{1,3}$ and $o_{1,2} \rightarrow o_{3,1} \rightarrow o_{2,3}$ and completion time of jobs are $C_1 = 15, C_2 = 11, C_3 = 11$ respectively.

3.2. Neighborhood structure

The most common neighborhood structure for JSP is defined by an interchange of processing sequence on the same machine. For makespan minimization, the longest path from the source node to the sink node with the maximum completion time is defined as critical path and it is proven that interchanges of operations on a critical path always leads to a feasible schedule and that only such interchanges can reduce the makespan immediately (van Laarhoven, et al., 1992). Therefore typical neighborhood consists of interchanges of adjacent operations on a critical path.

For the tardiness minimization, completion time of all jobs can affect the objective function. Thus, critical paths are defined for each job by the longest path from the source node to the sink node. As reducing the completion time of on-time jobs does not lead to immediately improving the total weighted tardiness, neighborhood is defined by reversal of
Yokoyama, Iizuka and Yamamoto, Journal of Advanced Mechanical Design, Systems, and Manufact, Vol.8, No.5 (2014)

(a) Disjunctive graph representation of the feasible schedule
(b) Corresponding Gantt chart of the feasible schedule

Fig. 3 A feasible schedule for the instance shown in Table 1 and Fig. 2

As completion time of $J_1$ and $J_2$ is later than due date, neighborhood is defined by a reversal of disjunctive arc $o_{1,2} \rightarrow o_{3,1}$, $o_{3,3} \rightarrow o_{1,3}$ or $o_{1,1} \rightarrow o_{2,1}$.

The neighborhood structure used by Kreipl (2000) permits reversal of multiple disjunctive arcs and provides wider range of neighborhood. However, in the method by Essafi et al. (2008), neighborhood is limited to reversal of single arc because the local search procedure is combined with genetic algorithm and thus limited amount of computational effort should be dedicated to the local search procedure. As our proposal method incorporates the local search procedure with newly proposed rule-based reconstruction procedure, limited neighborhood of single arc reversal is also used in the proposed method.

3.3. Iterative local search

If only improving moves are accepted with the neighborhood described above, the solution quickly reaches to one of local optima where no improving neighborhood is found. To continue local search process to find global optima, moves with no improvement must be permitted to escape from the local optimum. The previous studies suggest that small numbers of successive random moves are sufficient to escape from local optima for the tardiness criterion (Kreipl, 2000).

Although tabu search can also be used to prevent the convergence to local optima (Bontridder, 2005), random move strategy with genetic algorithm (Essafi, et al., 2008) outperforms their results. Therefore, the iterative local search procedure which alternately applies steepest descent moves and random moves is used in the proposed method described in the following.

4. Our proposed method

Our proposed method alternately applies conventional local search procedure and rule-based reconstruction procedure. The rule-based reconstruction procedure exploits priority rules to efficiently improve a feasible schedule obtained from the conventional local search procedure. If the current solution cannot be improved by the rule-based reconstruction procedure further, the conventional local search procedure is applied to improve the current solution. Typically, priority rules determine the whole operation sequences of the schedule and this prevents integration with local search methods as the sequences obtained by local search methods are completely overwritten by priority rules. Rule-based reconstruction procedure addresses this problem by selecting a time window and updating the operation sequences with priority rules only within the time window. Priority rules are given by linear combination of basic priority rules proposed by Tamura et al. (2013).

Selection of the time window and gains of priority rules is vital and should be dynamically optimized according to the given schedule. Therefore, they are optimized with genetic algorithm.

4.1. Rule-based reconstruction procedure

When a problem instance of JSP, a feasible schedule for the instance (denoted by the enumeration of starting time of
operations \(s_{i,j} (i = 1,2,\ldots,n, j = 1,2,\ldots,m)\) and a time window \([t_s, t_e]\) \((t_s < t_e)\) are given, the operations are divided into three categories \(P, L\) and \(S\). If \(s_{i,j} < t_s\) (i.e. starting time of operation \(o_{i,j}\) is earlier than \(t_s\)), \(o_{i,j}\) is included in \(P\). Similarly, \(o_{i,j}\) is classified to \(S\) if \(s_{i,j} > t_e\). Otherwise, operation \(o_{i,j}\) is an element of \(L\). If the instance is given by Table 1, the schedule is Fig. 3 and the time window is \([2, 8]\), operations are classified to \(P = \{o_{1,1}\}, L = \{o_{2,1}, o_{3,2}, o_{2,2}, o_{1,2}, o_{3,1}\}\) and \(S = \{o_{3,3}, o_{1,3}\}\) respectively as shown in Fig. 4.

![Fig. 4 Classification example of operations in the schedule shown in Fig. 3 by the time window of \([2, 8]\).](image)

In priority rule-based reconstruction procedure, the schedule is reconstructed to one of active schedules. In the reconstructed schedule, the sequences of operations included in \(P\) or \(S\) are retained as much as possible while sequences of the operations in \(L\) is determined with priority rules. To obtain the reconstructed schedule, GT algorithm by Giffler and Thompson (1960) is employed to acquire an active schedule and conflicts between operations are resolved by previous starting time or priority rules depending on their class. The following procedure obtains the reconstructed schedule.

1. Let \(A = \{o_{1,1}, o_{2,1}, \ldots, o_{n,1}\}\) be the set of all candidate operations
2. Let \(B = \emptyset\) be the set of all scheduled operations
3. Repeat step 4. to step 8. until starting time of all operations are determined and \(A = \emptyset\)
4. Find the operation \(o' \in A\) with the earliest possible completion time \(c'\) satisfying constraints with respect to any operation in \(B\)
5. Find the set of candidate operation \(C\) which is processed on the same machine as \(o'\) and the earliest possible starting time is less than \(c'\)
6. Choose the operation \(o''\) from \(C\). If no operation in \(C\) is classified to \(P\) and any operation in \(C\) is classified to \(L\), the operation is selected by priority rules from all operations included in \(C\) and belong to \(L\). Otherwise, the operation with the earliest possible starting time in the previous schedule is selected.
7. Starting time of \(o''\) in the reconstructed schedule is given by the earliest possible starting time of \(o''\)
8. Add \(o''\) to \(B\). If \(o''\) is not the final operation in the job, add the subsequent operation of \(o''\) to \(A\)

### 4.2. Linear combination of priority rules

Vepsalainen and Morton (1987) compares existing priority rules with tardiness criterion and Apparent Tardiness Cost (ATC) rule performs well for various problems on average. On the other hand, Tamura et al. (2013) shows that linear combination of priority rules can be more effective than conventional priority rules if the gains are optimized for a particular instance with makespan criterion. Our proposed method exploits priority rules to guide the local search procedure and thus a priority rule optimized for a particular instance is more useful than a priority rule which performs better in general. Therefore, linear combination of existing rules is used in our proposed method and its gains are optimized.

ATC rule consists of two simple priority rules and combines them according to the situation. First, slack per remaining processing time (S/RPT) of each operation is calculated and operation with the least value is selected. Slack is the amount of the time that the job can be delayed without breaking the deadline when all subsequent operations are processed with no idle time. If slack of candidate operations is large enough, weighted shortest processing time rule which prioritizes an operation with shorter processing time and larger weight value is employed. Therefore, slack per remaining processing time and weighted shortest processing time are used in the linear combination. ATC rule is typically used to generate a non-delay schedule. However, our proposed method searches for active schedule. As active schedules permit each machine to be idle waiting for a operation, possible machine idle time for a operation is considered so that an operation which leads to longer idle time can be penalized.

The priority rule used in our rule-based reconstruction procedure calculates priority value \(v_o\) for each candidate
operation by the following equation and prioritizes the operation with minimum priority value.

$$v_o = \alpha_1 WPT_o + \alpha_2 S/RPT_o + \alpha_3 PIM_o$$  \(5\)

where $\alpha_1$, $\alpha_2$ and $\alpha_3$ are gain parameters to be optimized and WPT, S/RPT and PIM are normalized weighted processing time, slack per remaining time and possible idling time of corresponding machine for each operation. For an operation $o_{i,j}$ whose earliest possible starting time is $t_i$ and the machine to be processed becomes idle at time $pm_{o_{i,j}}$, these values are calculated with the following equations using the earliest possible starting time $t_{o_{i,j}}$ of the operation $o_{i,j}$ and the earliest possible completion time $c'$ of $o'$ in the reconstruction procedure described above.

$$WPT_{o_{i,j}} = \frac{p_{o_{i,j}}}{u_i \max_{j\neq i} p_{i,j}}$$ \(6\)

$$S/RPT_{o_{i,j}} = \max \left\{ \min \left( \frac{d_i - t_{o_{i,j}} - \sum_{k=j+1}^{m} p_{i,k}}{\sum_{k=j+1}^{m} p_{i,k}}, 0.0 \right), 1.0 \right\}$$ \(7\)

$$PTM_{o_{i,j}} = \frac{c' - pm_{o_{i,j}}}{t_{o_{i,j}} - pm_{o_{i,j}}}$$ \(8\)

Note that slack per remaining time is truncated because most of efficient rules including ATC ignore negative slack value.

### 4.3. Integration with local search procedure

A feasible schedule obtained by the local search procedure can be reconstructed by the procedure described above if a time window and gains for the priority rule is given. Since the objective function is the minimization of the total weighted tardiness, we search for a schedule with less total weighted tardiness by the reconstruction procedure. Thus, the time window and the gains which minimizes the total weighted tardiness is required for the reconstruction. However, it is difficult to enumerate possible values and calculate the total weighted tardiness of the reconstructed schedules. Therefore, a time window and gains are optimized by genetic algorithm. The chromosome of genetic algorithm is represented by five dimensional vector of real numbers $[c, r, \beta_1, \beta_2, \beta_3]$. Parameters $c$ and $r$ determine a time window in which operations are prioritized by the linear combined priority rule. The time window is given by $t_c = C_{\max}(c - r)$ and $t_c = C_{\max}(c + r)$ where $C_{\max}$ is a makespan of the current schedule. Gain parameters of the linear combined rule are given by $\alpha_1 = \beta_1 - 0.5$, $\alpha_2 = \beta_2 - 0.5$, $\alpha_3 = \beta_3 - 0.5$.

The current solution is initialized with a random schedule. Rule-based reconstruction procedure and the iterative local search procedure is alternately applied. In the rule-based reconstruction procedure, the current schedule is updated only when a parameter set of time window and gains which reduces the total weighted tardiness of the current schedule is found. If no improving parameter set is found after a certain number of generations $r_{\text{max}}$, we assume that the current solution cannot be improved with rule-based reconstruction procedure and the iterative local search procedure is applied. Our iterative local search procedure is basically the same as the one used by Essafi et al. (2008). Starting from an initial solution, improvement and perturbation are performed alternately. Improvement consists of steepest descent search where the most improving move in the neighborhood is selected. Once the improvement algorithm reaches to one of local optima, perturbation is performed by random move. A move is randomly selected from the neighborhood regardless of whether it is improving or not. Random move is performed $t$ times where $t$ is chosen by a uniform random number $[t_{\text{min}}, t_{\text{max}}]$. As suggested by Essafi et al. (2008), these parameters are given by $t_{\text{min}} = 5$ and $t_{\text{max}} = 8$. The procedure stops if improvement procedure finds local optima $n_{\text{ad}}$ times. The best solution with the minimum total weighted tardiness during the procedure is retained and the current solution is replaced by the best solution. However, if the best solution has the same total weighted tardiness as that of the initial solution, we assume that the best solution is the same as the initial solution and the current solution is replaced with the local optimum obtained by the last improvement to avoid premature convergence.

The overall procedure of our proposed method is the following.

1. Initialize the current solution with a random schedule
2. Initialize $q$ chromosomes of five dimensional vectors with random uniform number $[0, 1]$  
3. Set $r = r_{\text{max}}$
4. If $r = 0$, go to step 8
5. Calculate the fitness value of each chromosome by performing rule-based reconstruction of the current schedule
6. If a schedule with less total weighted tardiness is found, update the current solution and set $r = r_{\text{max}}$
7. Decrement \( r \) and go back to step 4.
8. Perform the iterative local search \( n_{sd} \) times starting from the current solution.
9. Replace the current solution by the iterative local search procedure.
10. If terminal condition is met, stop the procedure. Otherwise, go back to step 3.

Step 3 to 7. perform local search through rule-based reconstruction procedure. Time window and rule gains are selected by the chromosomes and fitness value of each chromosome is given by total weighted tardiness of the reconstructed schedule. If the reconstructed schedule has the same critical path as the previous schedule, the fitness value are doubled to penalize premature convergence. Crossover operation are defined by uniform crossover of each elements of the five dimensional vector while mutation operation randomly selects one element and changes it with uniform random number \([0, 1]\). If an improving schedule is found by the procedure above, the current schedule is updated. This local search with rule-based reconstruction proceeds until improving chromosome is not found in \( r_{max} \) successive generations. \( n_{sd} \), \( g \) and \( r_{max} \) defines computational efforts dedicated to each procedure. The larger \( n_{sd} \) spends more time on the conventional local search procedure while larger \( g \) and \( r_{max} \) take more time to rule-based reconstruction procedure.

5. Computational experiments

In order to verify the effectiveness of our proposed method and the priority rule-based reconstruction procedure, computational experiments are performed. Our proposed method is implemented and executed on a personal computer with 3.3GHz Core i5 processor and 4GB RAM.

Benchmark instances proposed by Pinedo and Singer (1999) and Essafi et al. (2008) are used to evaluate the quality of heuristic methods. These instances are derived from benchmark instances for makespan criterion with additional weights and days due to jobs. For 20% of the total jobs, \( w_i = 4 \) is given and \( w_i = 2 \) is given for 60% of the jobs. Remaining 20% of the jobs have \( w_i = 1 \). Due date are given by \( d_i = f_i \cdot \sum_{j} P_{i,j} \) for all jobs where \( f \) is a tightness factor of due dates. Three values of tightness factor \( f = 1.3 \), \( f = 1.5 \) and \( f = 1.6 \) are considered. Benchmark by Pinedo and Singer (1999) consists of 22 instances with 10 machines and 10 jobs. Some instances have larger number of jobs in the original benchmark set and last jobs are eliminated in this benchmark set. Optimal values of total weighted tardiness for these instances are calculated with branch and bound method by Singer and Pinedo (1998). However, better solution is found by subsequent research (Essafi, et al., 2008) and it is considered that the branch and bound algorithm is stopped before finding the optimal solution. 40 benchmark instances proposed by Essafi et al. (2008) include larger instance with up to 300 operations. So far, optimal values of total weighted tardiness for these instances have not been known due to their larger size.

5.1. Performance of rule-based reconstruction procedure

To evaluate the performance of our priority rule-based reconstruction procedure combined with conventional local search, the proposed method is performed on different parameter sets to change the computational effort dedicated to the rule-based reconstruction procedure. Medium sized benchmark set proposed by Essafi et al. (2008) is used and the terminal condition is set to the total elapsed time of 300 seconds. The number of chromosomes in rule-based reconstruction procedure is fixed at \( g = 100 \). As suggested by Essafi et al. (2008), the maximum iteration number of conventional local search procedure \( n_{sd} \) is given by a uniform random number \([n_{sd}^{min}, n_{sd}^{max}]\) and \( n_{sd}^{min} = 50 \), \( n_{sd}^{max} = 80 \) is given. The number of maximum generations to find rule gains \( r_{max} \) is set by either \( r_{max} = 0 \) or \( r_{max} = 10 \). Conventional local search method is represented by \( r_{max} = 0 \) where no improvement with rule-based reconstruction is performed and the rule reconstruction procedure is added by setting \( r_{max} = 10 \). Binary tournament selection and uniform crossover are used in the genetic algorithm to find a set of time window and rule gains. Mutations are performed by updating one element of a chromosome by a uniform random number. Probability of crossover and mutation is set to 0.8 and 0.1 respectively. The proposed method is performed on benchmark instances 10 times independently with each of the parameter to collect the total weighted tardiness of a obtained solution. Relative error is calculated as \( (TWT_0 - TWT_{10})/TWT_{10} \) where \( TWT_{10} \) is the total weighted tardiness obtained with \( r_{max} = 10 \) and \( TWT_0 \) is the total weighted tardiness obtained with \( r_{max} = 0 \). The relative error of the mean total weighted tardiness over 10 trials is denoted by \( dev_{in} \) and the relative error of the best solution over 10 trials are denoted by \( dev_{br} \) respectively.

Table 2 shows the results. On average, the rule-based reconstruction procedure yields better solution for 65 out of 140 instances while only conventional local search is better for 12 instances. Comparing the best result over 10 runs, the rule based reconstruction works well for 42 instances and has negative effect for 17 instances. For the remaining instances
Table 2 Relative error of the proposed method between the parameters of $r_{\text{max}} = 10$ and $r_{\text{max}} = 0$.

| Instance | $n$ | $m$ | $f = 1.3$ | $f = 1.5$ | $f = 1.6$ |
|----------|----|----|------------|------------|------------|
|          |    |    | $d_{\text{evn}}$ | $d_{\text{evb}}$ | $d_{\text{evn}}$ | $d_{\text{evb}}$ |
| LA01     | 10 | 5  | 0          | 0          | 0          |
| LA02     | 10 | 5  | 0          | 0          | 0          |
| LA03     | 10 | 5  | 0          | 0          | 0          |
| LA04     | 10 | 5  | 0          | 0          | 0          |
| LA05     | 10 | 5  | 0          | 0          | 0          |
| LA06     | 15 | 5  | 0.005      | 0          | 0          |
| LA07     | 15 | 5  | 0          | 0          | 0.001     |
| LA08     | 15 | 5  | 0          | 0          | 0          |
| LA09     | 15 | 5  | 0          | 0          | 0.001     |
| LA10     | 15 | 5  | 0          | 0          | 0          |
| LA11     | 20 | 5  | 0.007      | -0.005     | 0.001     |
| LA12     | 20 | 5  | 0.008      | 0.009      | 0.009     |
| LA13     | 20 | 5  | 0.007      | 0.010      | 0.006     |
| LA14     | 20 | 5  | -0.006     | -0.004     | 0.001     |
| LA15     | 20 | 5  | 0.003      | -0.002     | 0.005     |
| LA16     | 10 | 10 | 0          | 0          | 0          |
| LA17     | 10 | 10 | 0          | 0          | 0          |
| LA18     | 10 | 10 | 0          | 0          | 0          |
| LA19     | 10 | 10 | 0          | 0          | 0          |
| LA20     | 10 | 10 | 0          | 0          | 0          |
| LA21     | 15 | 10 | 0.018      | 0          | 0.004     |
| LA22     | 15 | 10 | 0.013      | 0.001      | 0.063     |
| LA23     | 15 | 10 | 0.022      | 0          | 0.001     |
| LA24     | 15 | 10 | 0.043      | 0.013      | 0.007     |
| LA25     | 15 | 10 | 0.006      | 0.012      | -0.002    |
| LA26     | 20 | 10 | 0.068      | -0.009     | 0.042     |
| LA27     | 20 | 10 | 0.001      | 0.004      | 0.028     |
| LA28     | 20 | 10 | 0.018      | -0.009     | 0.012     |
| LA29     | 20 | 10 | 0.001      | -0.021     | 0.010     |
| LA30     | 20 | 10 | 0.019      | 0.004      | 0.014     |
| LA31     | 30 | 10 | 0.078      | -0.009     | 0.064     |
| LA32     | 30 | 10 | 0.077      | 0.067      | 0.098     |
| LA33     | 30 | 10 | 0.174      | 0.083      | 0.165     |
| LA34     | 30 | 10 | 0.094      | 0.066      | 0.106     |
| LA35     | 30 | 10 | 0.100      | 0.057      | 0.112     |
| LA36     | 15 | 15 | 0.004      | 0.004      | -0.037    |
| LA37     | 15 | 15 | -0.010     | 0.001      | 0.035     |
| LA38     | 15 | 15 | 0.027      | 0.044      | -0.073    |
| LA39     | 15 | 15 | 0.001      | -1.000     | 0          |
| LA40     | 15 | 15 | 0.020      | 0.023      | -0.016    |

the performance is same regardless of rule-based reconstruction procedure. On small instances, the performances of the two methods (i.e. $r_{\text{max}} = 10$ and $r_{\text{max}} = 0$) are roughly equivalent while the rule-based reconstruction procedure works well on larger instances. This result verifies that rule-based reconstruction improves overall performance of conventional local search. The rule-based reconstruction procedure can modify the sequences of operations on multiple machines. This feature helps escaping from local optima of conventional local search and thus average performance of the procedure is increased.

5.2. Comparison with existing methods

Our proposed method is compared with GLS (Essafi, et al., 2008) which is one of the best existing methods on relatively small benchmark instances proposed by Pinedo and Singer (1999). The maximum computation time is set to 10 seconds while the GLS is performed for 18 seconds on a personal computer with 2.8GHz processor. The result is shown in Table 3. For each instance, column Opt. shows results from the branch and bound method by Singer and Pinedo (1998) and column GLS shows the best and mean values on 10 independent runs reported by Essafi et al. (2008). Our proposed method is also performed on each instance for 10 times independently. The parameter which is confirmed to be efficient in the former experiment is used for our proposed method. The number of chromosomes is given by $g = 100$
and the number of maximum generation to find rule gains are fixed at $r_{\text{max}} = 10$ from the former experiment. The maximum iteration number of conventional local search procedure $n_{\text{sd}}$ is given by a uniform random number $[n_{\text{min}}^{sd}, n_{\text{max}}^{sd}]$ and $n_{\text{sd}}^{min} = 50, n_{\text{sd}}^{max} = 80$ is given as suggested by Essafi et al. (2008). We estimate mean values of total weighted tardiness of the proposed method for each instance and test significant difference from the reported mean value of GLS since only mean and best values of GLS are available. Our proposed method significantly outperforms GLS on 7 out of 66 instances while GLS significantly performs better on 10 instances with significance level of 5%. The result with significant difference are written in bold. Therefore, the performance of our proposed method is competitive with GLS on the benchmark set. The explanation is that rule-based reconstruction procedure requires large amount of computational efforts to find appropriate sets of time window and rule gains while this procedure can find the solution which cannot easily obtained with the conventional local search and thus the allowed computational time is not enough to make use of the strengths of both methods.

To evaluate performance of our proposed method with sufficient computational time, medium sized benchmark instances proposed by Essafi et al. (2008) is tested. This benchmark set includes more difficult instances with up to 30 jobs and 10 machines. LA16-LA20 are overlapped in the previous benchmark. Our proposed method is performed on each instance in 30 seconds while GLS is performed on these instances for 200 generations and their computational time is not presented. The parameters of our proposed method are the same as the previous experiment and both methods are performed for each instance 10 times. The results are presented in Table 4. Both methods yield optimal solution for LA16-LA20 in every trial. This implies that computational time is not sufficient for the both methods in the previous experiment. On small instances with $n \leq 10$ and $m \leq 10$ including LA16-LA20, both methods are equally efficient. On large instances, however, our proposed method is considerably more efficient exceeding the best solution found by GLS on 38 instances. Estimated mean values of total weighted tardiness acquired by the proposed method are compared to reported mean values of GLS as with smaller benchmark problems. The result shows that our proposed method outperforms GLS in 67 instances and that GLS performs better in 3 instances with significance level of 5%. Significantly different results are shown in bold in Table 4. This is because the selection procedure in GLS does not consider the diversity of solutions and thus premature convergence can be occurred. On the other hand, our proposed method applies the rule-based reconstruction procedure which can impose a substantial change to the temporal solution and enable to search diverse solution by local search procedure.

### 6. Conclusion

We proposed priority rule-based reconstruction method integrated with the conventional local search method for...
Table 4 Comparison results of our proposed method with GLS on medium sized benchmark instances

| Instance | n | m | \( f = 1.3 \) Proposal Mean | \( f = 1.3 \) GLS Mean | \( f = 1.5 \) Proposal Mean | \( f = 1.5 \) GLS Mean | \( f = 1.6 \) Proposal Mean | \( f = 1.6 \) GLS Mean |
|----------|---|---|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| LA01     | 10| 5 | 2299            | 2299            | 1610            | 1610            | 1230            | 1230            |
| LA02     | 10| 5 | 1762            | 1762            | 1028            | 1028            | 695             | 695             |
| LA03     | 10| 5 | 1951            | 1951            | 1280            | 1280            | 1024            | 1024            |
| LA04     | 10| 5 | 1917            | 1917            | 1277            | 1277            | 1029            | 1029            |
| LA05     | 10| 5 | 1878            | 1878            | 1205            | 1205            | 877             | 877             |
| LA06     | 10| 5 | 5810            | 5810            | 4592            | 4592            | 4089            | 4089            |
| LA07     | 10| 5 | 5769            | 5769            | 4471.2          | 4470            | 3854.2          | 3851            |
| LA08     | 10| 5 | 5475            | 5475            | 4034            | 4034            | 3400            | 3400            |
| LA09     | 10| 5 | 5608            | 5608            | 4419            | 4421            | 3811            | 3811            |
| LA10     | 10| 5 | 6618            | 6618            | 5148            | 5148            | 4503            | 4503            |
| LA11     | 10| 5 | 12131.4         | 12085           | 10171           | 10054           | 9259.6          | 9232            |
| LA12     | 10| 5 | 10608.4         | 10542           | 9025.9          | 8956            | 8222.6          | 8178            |
| LA13     | 10| 5 | 11555.8         | 11502           | 9600.4          | 9594            | 8732.1          | 8672            |
| LA14     | 10| 5 | 13277.2         | 13177           | 11436           | 11394           | 10601.6         | 10549          |
| LA15     | 10| 5 | 12415.6         | 12376           | 10409.7         | 10435           | 9429.1          | 9328            |
| LA16     | 10| 10| 1169            | 1169            | 166             | 166             | 0               | 0               |
| LA17     | 10| 10| 899             | 899             | 260             | 260             | 65              | 65              |
| LA18     | 10| 10| 929             | 929             | 34              | 34              | 0               | 0               |
| LA19     | 10| 10| 948             | 948             | 21              | 21              | 0               | 0               |
| LA20     | 10| 10| 805             | 805             | 0               | 0               | 0               | 0               |

JSP with total weighted tardiness criterion. Our proposed method was compared with state-of-the-art approach for JSP with tardiness criterion and the result shows that the proposed method is superior given that sufficient computational time is allowed. The rule-based reconstruction procedure modifies the operation sequences within an arbitrary selected time window and this enables to deal with the solution representation typically used in local search methods. In the reconstruction procedure, linear combination of simple priority rules is adopted. The gains are optimized by genetic algorithm along with the time window to get a better solution and the number of chromosomes defines the computational effort dedicated to the reconstruction procedure. This reconstruction procedure and conventional local search procedure are alternately applied to the temporal solution. Experiment results showed the rule-based reconstruction procedure improves the overall performance. Finally, our proposal method was applied to the benchmark instances proposed by Essafi et al. (2008) and the proposal method yielded considerably better solutions on these instances than their efficient GLS method.

To improve our method further, several perspectives can be considered. First, components of the linear combined priority rule are arbitrary selected from relatively simple priority rules and more sophisticated rules from recent studies can improve the performance of the rule-based reconstruction procedure. Optimization of the gains of linear combined rule...
by genetic algorithm becomes difficult with the increasing number of its component. Therefore, components of the liner combined rule should be carefully selected. Additionally, the proposed method can be used for other criteria with appropriate priority rules. For example, makespan minimization can be considered by exchanging due date related priority rules with priority rules which regards total remaining operation time or number of remaining operations. Finally, the amount of computational effort dedicated to rule-based reconstruction procedure needed for maximum overall performance is still unknown. In the situation that the rule-based reconstruction is likely to improve the quality of the temporal solution, extensive search of the time window and rule gains should be performed and iteration number of local search should be increased otherwise.

References

Bierwirth, C., A generalized permutation approach to job shop scheduling with genetic algorithms, Operations-Research-Spektrum, Vol.17, No.2-3 (1995), pp.87-92.
Bontridder, K., Minimizing total weighted tardiness in a generalized job shop, Journal of Scheduling, Vol.8, No.6 (2005), pp.479-496.
Essafi, I., Mati, Y. and Dauzère-Pérès, S., A genetic local search algorithm for minimizing total weighted tardiness in the job-shop scheduling problem, Computers & Operations Research, Vol.35, No.8 (2008), pp.2599-2616.
Gao, L., Zhang, G., Zhang, L. and Li, X., An efficient memetic algorithm for solving the job shop scheduling problem, Computers & Industrial Engineering, Vol.60, No.4 (2011), pp.699-705.
Giffler, B. and Thompson, G. L., Algorithms for solving production-scheduling problems, Operations Research, Vol.8, No.4 (1960), pp.487-503.
Hasan, S., Sarker, R., Essam, D. and Cornforth, D., Memetic algorithms for solving job-shop scheduling problems, Memetic Computing, Vol.1, No.1 (2009), pp.69-83.
Kreipl, S., A large step random walk for minimizing total weighted tardiness in a job shop, Journal of Scheduling, Vol.3, No.3 (2000), pp.125-138.
Nowicki, E. and Smutnicki, C., A fast taboo search algorithm for the job shop problem, Management Science, Vol.42, No.6 (1996), pp.797-813.
Panwalkar, S. S. and Iskander, W., A survey of scheduling rules, Operations Research, Vol.25, No.1 (1977), pp.45-61.
Pinedo, M. and Singer, M., A shifting bottleneck heuristic for minimizing the total weighted tardiness in a job shop, Naval Research Logistics (NRL), Vol.46, No.1 (1999), pp.1-17.
Qing-dao-er-ji, R. and Wang, Y., A new hybrid genetic algorithm for job shop scheduling problem, Computers & Operations Research, Vol.39, No.10 (2012), pp.2291-2299.
Singer, M. and Pinedo, M., A computational study of branch and bound techniques for minimizing the total weighted tardiness in job shops, IIE Transactions, Vol.30, No.2 (1998), pp.109-118.
Taillard, E. D., Parallel taboo search techniques for the job shop scheduling problem, ORSA Journal on Computing, Vol.6, No.2 (1994), pp.108-117.
Tamura, Y., Yamamoto, M., Suzuki, I. and Furukawa, M., Acquisition of dispatching rules for job-shop scheduling problem by artificial neural networks using PSO, Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol.17, No.5 (2013), pp.731-738.
van Laarhoven, P. J. M., Aarts, E. H. L. and Lenstra, J. K., Job shop scheduling by simulated annealing, Operations Research, Vol.40, No.1 (1992), pp.113-125.
Vepsalainen, A. P. J. and Morton, T. E., Priority rules for job shops with weighted tardiness costs, Management Science, Vol.33, No.8 (1987), pp.1035-1047.
Zhang, C., Li, P., Guan, Z. and Rao, Y., A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem, Computers & Operations Research, Vol.34, No.11 (2007), pp.3229-3242.
Zhang, R., Song, S. and Wu, C., A dispatching rule-based hybrid genetic algorithm focusing on non-delay schedules for the job shop scheduling problem, The International Journal of Advanced Manufacturing Technology, Vol.67, No.1-4 (2013), pp.5-17.
Zhou, H., Cheung, W. and Leung, L. C., Minimizing weighted tardiness of job-shop scheduling using a hybrid genetic algorithm, European Journal of Operational Research, Vol.194, No.3 (2009), pp.637-649.

[DOI: 10.1299/jamdsm.2014jamdsm0073] © 2014 The Japan Society of Mechanical Engineers