Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Scheduling algorithm with controllable train speeds and departure times to decrease the total train tardiness

Omid Gholami[*] **and Yuri N. Sotskov**

United Institute of Informatics Problems, Surganov Str 6, Minsk 220012, Belarus

C H R O N I C L E

Article history:
Received July 2 2013
Received in revised format September 7 2013
Accepted November 23 2013
Available online November 17 2013

Keywords:
Train timetabling
Job-shop scheduling
Makespan
Total tardiness
Dispatching rules

A B S T R A C T

The problem of generating a train schedule for a single-track railway system is addressed in this paper. A three stage scheduling is proposed to reduce the total train tardiness. We derived an appropriate job-shop scheduling algorithm called DR-algorithm. In the first stage, by determining appropriate weights of the dispatching rules, a pre-schedule is constructed. In the second stage, on the basis of the pre-schedule, the departure times of the trains are modified to reduce the number of conflicts in using railway sections by different trains. In the third stage, a train speed control helps the scheduler to change the trains' speeds in order to reduce the train tardiness and to reach other objectives. The factual train schedule is based on the modified train speeds and on the modified departure times of the trains. The experimental running of the DR-algorithm on the benchmark instances showed this algorithm can solve train scheduling problems in a close to optimal way. In particular, the total train tardiness was reduced about 20% due to controlling train speeds and the departure times of the trains.

1. Introduction

Railway traffic has been essentially increased in the last decades (see Lusby et al., 2011 for a survey). The usage of the railroad systems grows for the passenger and freight transportation. Safety and low cost of the railway transportation attract people to use trains more, which causes railway authorities to make a maximal usage of the existing railways. The train speeds and the number of trains moving on a railway system are increasing. As a consequence, a delay of a train arises from time to time. A train delay creates a lot of problems for the railway company including dissatisfaction of passengers about the quality of services and a financial damage associated with excessive train delays. The railway companies are forced to pay penalties to passengers for their delays. To make the usage of a railway system more affective, several approaches for solving the train scheduling problems have been proposed in the last decade. Zhou and Zhong (2007) introduced a resource-constrained project scheduling used for a single-track timetabling problem. Railway segments and stations were considered as limited resources. Such a problem is solved by a branch and bound (B&B) algorithm that segment and station capacity constraints were used as a lower bound. The authors considered a lower bound for

* Corresponding author. Tel:+375 29251 3440
E-mail: gholami@iaumah.ac.ir (O. Gholami)

doi: 10.5267/j.ijiec.2013.11.002

a less train delay. An upper bound was constructed via a beam search heuristic. A B&B algorithm was also used for a mixed integer non-linear mathematical programming reported by Kraay et al. (1991) with presenting the computational results for a 102 mile stretch of track interlinking 13 sidings with 22 trains in common. A train pacing problem has been considered, where a speed profile for each train has to be determined. Jovanovic and Harker (1991) proposed mixed integer programming, which is similar to a flow-shop scheduling problem. Two types of the variables were used in the proposed algorithm. The binary variables were used for ordering pairs of trains. The other variables were the continuous variables used for selecting the departure times of the trains. The proposed algorithm could solve the problem with 24 railway segments and with 100 trains. Szpigel (1973) was the first who identified the similarities between a job-shop scheduling problem and a train scheduling problem in the case of a single-track railway. The former was solved by Szpigel (1973) using a B&B algorithm, the initial linear programming excluding order constraints. Branching was required if the current solution contains trains which were in a conflict (i.e., when trains turn out to be on the same railroad section at the same time). The objective was to minimize the weighted sum of the train transit times. The computational results for 5 single-track sections and 10 trains have been reported.

The same problem was considered by Carey and Lockwood (1995) via binary mixed integer programming similarly to that considered by Jovanovic and Harker (1991). Temporal constraints were identical to those used by Szpigel (1973). The objective was to minimize the deviation from the ideal arrival times and the ideal departure times for all the trains to be scheduled. Mladenovic and Cangalovic (2007) used job-shop scheduling problem as a way to solve the train scheduling problem where a route was interpreted as follows. The route is a sequence of facilities the train must cross from the originating station to the destination. Assuming that the train trips are jobs to be scheduled, which require the elements of infrastructure as restricted resources, it was made by the mapping of the initial problem into a special case of a job-shop scheduling problem. In order to solve the job-shop scheduling problem, a constraint programming approach has been developed. A support to fast finding a good schedule was offered by an original separation and a bound-and-search heuristic. To improve the time performance, a surrogate objective function was used which had a smaller domain than the actual objective function.

There are variety of algorithms to schedule jobs in a job-shop scheduling problem like the shifting bottleneck algorithm (Adams et al., 1998) that tries to find the most bottleneck machine in each irritation. Operations on that specific machine are scheduled as a single machine problem. The procedure is continued for all remaining machines in M or it is stoped when there is no machine with lateness for operations. A tabu search algorithm is a local search one used for job-shop scheduling; Glover (1989). A tabu search algorithm adopts a local search approach with a ’memory’ implemented as a ’tabu-list’ of moves which have been made in the recent history of the search, and which are forbidden (tabu) for a certain number of iterations which follow. Simulated annealing is a local search meta-heuristic for the optimization problems. Simulated annealing tries to escape local optima by hill-climbing techniques. At each step, the simulated annealing algorithm changes the current solution by a random solution and used for scheduling by Van Laarhoven (1992). Shafia et al. (2010) tried to reduce the tardiness of operations (equivalent to trains latecy) by developing a robust job-shop scheduler, which has the capability of handling the perturbation that exists among almost all input parameters. The aim of developed algorithm was, by small alteration in the input parameters reduces the latency. A simulated annealing algorithm has been proposed to find near optimal solutions in a reasonable time. Ghoseiri et al. (2004) developed a multi-objective optimization model for the train scheduling problem. They considered both single and multiple track railway systems. Their objective is defined as lowering the fuel consumption cost and minimizing total passenger time. First, they solved the problem by a Pareto algorithm, then they tried to use a multi-objective optimization to tune the results. There are some other reports about multi-objective optimization like Naderi-Beni et al. (2012) that tries to reduce two objectives of weighted mean tardiness and makespan. This model can be suitable to distinguish between passenger trains that tardiness is the main critaria in scheduling and fright trains that the makespan is most important factor. Dorfman and Medanic (2004) used a discrete-event model to schedule the traffic on a railway network. They claimed that it was an efficient technique with respect to the time needed to travel criteria. Burdett and Kozan (2010) made a relationship between flexible job-shop (with parallel machines) and train scheduling. They used a disjunctive graph to model the train scheduling problem. Pacciarelli and Pranzo (2001) used an extension of the disjunctive graph model. A tabu search algorithm was used to solve multi-track railway scheduling problem. A greedy heuristic was proposed by Cai and Goh (1994) for the train scheduling in a single-track railway. There is a limitation in their algorithm because they assumed that all trains moved in the same direction must have the same speed and terminating siding.

2. Problem setting

The problem of a timetable generation has to be solved at a tactical level of the railway planning process; Lusby et al. (2011). In a job-shop approach to train scheduling, trains and railroad sections are synonymous with jobs and machines, respectively. So, in the following setting of the optimization problem, job-shop terms are given in parenthesis after railway terms (or vice versa).
Let a set of railroad sections (machines) $M = \{\mu_1, \mu_2, \ldots, \mu_m\}$ and a set of trains (jobs) $J = \{\tau_1, \tau_2, \ldots, \tau_n\}$ be given before scheduling. For each train $\tau_i \in J$, it is given an ordered set (a route) of the railroad sections (machines), which have to be visited by train τ_i. To be more precise, a sequence of the job operations on the corresponding machines is given as follows:

$$Q_i = (o_{i1}^{\mu(i1)}, o_{i2}^{\mu(i2)}, \ldots, o_{in_i}^{\mu(in_i)}). \tag{1}$$

Hereafter, an operation $o_{ij}^{\mu(ij)}$ is regarded as the movement of a train $\tau_i \in J$ across a railroad section $\mu(ij) = \mu_k \in M = \{\mu_1, \mu_2, \ldots, \mu_m\}$. Preemption of any operation $o_{ij}^{\mu(ij)}$ is not allowed. Like in a classical job-shop problem, Tanaev et al. (1994), any machine $\mu_k \in M$ can be used to process a job $\tau_i \in J$ at most once according to the given route (1), i.e., any two different operations $o_{ir}^{\mu(ir)}$ and $o_{is}^{\mu(is)}$, $r \neq s$, of the same job (train) $\tau_i \in J$ have to be processed by the different machines (are movements across the different railroad sections). Due to this condition, an operation $o_{ir}^{\mu(ir)}$ may be identified with the corresponding machine $\mu(ir) = \mu_k \in M$, which has to process operation $o_{ir}^{\mu(ir)}$. Let a positive number p_{ij} denote the time required for train $\tau_i \in J$ to pass through the railroad section $\mu(ij) \in M$. In the other words, number p_{ij} denotes the processing time of operation $o_{ij}^{\mu(ij)}$ of the job (train) $\tau_i \in J$ processed on machine $\mu(ij)$.

Let a non-negative number r_i denote the earliest departure time of the train (the release time of the job) $\tau_i \in J$ from the original station in the given route (1). Let a positive number d_i denote the official arrival time of the train (or due date for the completion time of the job) $\tau_i \in J$ to the terminal station in the route (1). A non-negative weight w_i is associated with the train (job) $\tau_i \in J$ reflecting its importance. Let C_i denote the completion time of the job (the arrival time of the train) $\tau_i \in J$. The main objective under consideration is to find a train (job) schedule minimizing the following sum

$$\sum_{i=1}^{n} w_i T_i \textit{ of the weighted tardiness } T_i = \max\{0, C_i - d_i\} \tag{2}$$

for all trains (jobs) $\tau_i \in J$. According to the three-field notations used for machine-scheduling problems (see Graham et al., 1979), the above scheduling problem is denoted as $J \mid r_i \mid \sum w_i T_i$. The minimal expected completion time for each job (train) can be calculated as $c_{in_i} = \sum_{j=1}^{n_i} p_{ij}$. Different jobs (trains) may need the same machine (railroad section) at the same time and so they must wait until the machine (railroad section) be free, therefore the completion (arrival) time C_i of jobs (trains) may be different from (larger than) the minimal expected completion time c_{in_i}. In train scheduling, we consider the due date d_i equals to the minimal expected completion time c_{in_i}, i.e., $d_i = c_{in_i}$. The tardiness T_i is calculated as the difference between minimum completion time c_{in_i} and the real completion time C_i of the trains: $T_i = C_i - c_{in_i}$. Along with criterion (2), we consider criterion $\sum w_i C_i$ of minimizing the weighted sum of the job completion times

$$\sum_{i=1}^{n} w_i C_i \tag{3}$$

and criterion $C_{\max}$ of minimizing the makespan

$$\max_{i=1}^{n} C_i = \max\{C_i : \tau_i \in J\}. \tag{4}$$

Note that criterion (2) is mainly used for the passenger transportation, while criteria (3) and (4) are more important for the freight transportation. For a railway company, all three problems $J \mid r_i \mid \sum w_i T_i$, $J \mid r_i \mid \sum w_i C_i$, and $J \mid r_i \mid C_{\max}$ are useful to be solved at a tactical level of the railway planning; see Lusby et al. (2011). We remind that a regular criterion means to minimize a real-valued function $F(C_1, C_2, \ldots, C_n)$ that is non-decreasing for all the arguments $C_1, C_2, \ldots, C_n$. It is clear that the above three criteria $\sum w_i T_i = \sum_{i=1}^{n} w_i T_i$, $\sum w_i C_i = \sum_{i=1}^{n} w_i C_i$, and $C_{max} = \max\{C_i : \tau_i \in J\}$ are regular. A feasible schedule is called semi-active if no operation $o_{ij}^{\mu(ij)} \in \bigcup_{i=1}^{n} Q_i$ can be started earlier without increasing the starting time of another operation or altering the operation sequence processed on any machine from set M. For a regular criterion, at least one optimal schedule is semi-active (see Graham et al., 1979; or Tanaev, et al., 1994). Priority dispatching rules have been studied in the literature for several decades since they are widely used for different scheduling problems like the job-shop scheduling problem arising in the real world (see Haupt, 1989; Muth & Thompson, 1963; Panwalkar & Iskander, 1977; Tanaev et al., 1994). However, the conclusion of many years of research is that no priority dispatching rule performs better than the other ones tested for a rather wide class of scheduling problems. So, several researchers developed tools to discover effective priority dispatching rules automatically (see Abdolzadeh & Rashidi, 2010; Dorndorf & Pesch, 1995; Gabel & Riedmiller, 2007; Geiger et al., 2006; Li & Shi, 1994).

In this paper, we develop a weighted mixed priority dispatching rule scheduler (we call it DR-algorithm) for solving the classical job-shop scheduling problems $J \mid r_i \mid \sum w_i T_i$, $J \mid r_i \mid \sum w_i C_i$, and $J \mid r_i \mid C_{\max}$. The rest of the paper is organized as follows. We use a mixed graph to model the job-shop scheduling problem (Section 3). A three-stage strategy is proposed to reduce the total job tardiness (the total train delay time). In the first stage, the jobs (trains) are pre-scheduled using the DR-algorithm (Sections 4 and 5). In the next stage, the tardiness of each job (train) is measured. The algorithm tries to modify the departure time of the trains (the due date of the job completion) due to information obtained at the pre-scheduling stage in order to decrease the total train (job) tardiness (Subsection 6.1). In the third stage, a process controlling policy is used to improve the quality of scheduling and to make the final train schedule (Subsection 6.2). An illustrative example is given in Subsection 6.3. Computational results are discussed in Section 7. In Section 8, concluding remarks and perspectives are given. In what follows, we use the survey (Lusby et al., 2011) and the monographs (Tanaev et al., 1994; Thulasiraman & Swamy, 1992) for terminology on train timetabling, scheduling theory and graph theory, respectively.

3. A mixed graph model for the job-shop scheduling problem

We use a mixed graph $G = (Q, A, E)$ to model the scheduling problems $J \mid r_i \mid \sum w_i T_i$, $J \mid r_i \mid \sum w_i C_i$ and $J \mid r_i \mid C_{\max}$ under consideration. Mixed graph G allows us to present a problem data and to describe algorithms for solving the job-shop problem (Tanaev et al., 1994). In such a mixed graph $G = (Q, A, E)$, the vertex set $Q = \{\bigcup_{i=1}^{n} Q_i\} \bigcup \{o, *\}$ is a union of two dummy operations (o and $*$) and the set of all operations to be processed on machines M . The dummy operation o determines the starting time $t = 0$ of a schedule to be constructed. The dummy operation $*$ determines the completion time of the last operation in a schedule. The positive weight p_{ij} (the operation processing time) is prescribed to the vertex (to the operation) $o_{ij}^{\mu(ij)}$, where $\tau_i \in J$, $j \in \{1,2,\ldots,n_i\}$. Arc set A of the mixed graph G defines the precedence constraints implied by the ordered sets Q_i, $\tau_i \in J$, i.e., inclusion $(o_{ij-1}^{\mu(ij-1)}$, $o_{ij}^{\mu(ij)}) \in A$ holds for each index $j \in \{2,3,\ldots,n_i\}$ and for each index $i \in \{1,2,\ldots,n_i\}$. Arc set A defines also the preceding of the dummy operation o to the first operation $o_{i1}^{\mu(i1)}$ for each job $\tau_i \in J$, i.e., inclusion $(o, o_{i1}^{\mu(ij)}) \in A$ holds for each job (train) $\tau_i \in J$. The non-negative weight r_i (the job release time) is prescribed to the arc $(o, o_{i1}^{\mu(i1)}) \in A$, where $\tau_i \in J$. Edge set $E = \bigcup_{k=1}^{m} E^k$ of the mixed graph G defines the machine constraints. At any time, each machine $\mu_k \in M$ can process at most one operation from the set Q^k of all operations $o_{i1}^{\mu_k} \in Q$, which have to be processed on the machine $\mu_k \in M$. If both operations $o_{ij}^{\mu(ij)}$ and $o_{uv}^{\mu(uv)}$ belong to the set Q^k, i.e., the equalities $\mu(ij) = \mu_k = \mu(uv)$ hold, then edge $[o_{ij}^{\mu(ij)}, o_{uv}^{\mu(uv)}] = [o_{ij}^{\mu_k}, o_{uv}^{\mu_k}]$ has to belong to the set E^k. Thus, each vertex-induced subgraph $G^k = (Q^k, \varnothing, E^k)$ of the mixed graph $G = (Q, A, E)$ is a complete graph. Let $\Gamma(G)$ denote a set of all directed graphs $G_r = (Q, A \cup A_r, \varnothing)$ generated by the mixed graph $G = (Q, A, E)$ via orienting all edges E. In the directed graph $G_r = (Q, A \cup A_r, \varnothing) \in \Gamma(G)$, each edge $[o_{ij}^{\mu(ij)}, o_{uv}^{\mu(uv)}] \in E$ is replaced either by arc $(o_{ij}^{\mu(ij)}, o_{uv}^{\mu(uv)}) \in A_r$ (i.e., operation $o_{ij}^{\mu(ij)}$ has to be processed before operation $o_{uv}^{\mu(uv)}$ on machine $\mu(ij) = \mu(uv) = \mu_k \in M$) or by arc $(o_{uv}^{\mu(uv)}, o_{ij}^{\mu(ij)}) \in A_r$ (i.e., operation $o_{uv}^{\mu(uv)}$ has to be processed before operation $o_{ij}^{\mu(ij)}$ on machine $\mu(ij) = \mu(uv) = \mu_k \in M$). Thus, each schedule existing for the problem $J \mid r_i \mid \sum w_i T_i$ determines a circuit-free directed graph belonging to the set $\Gamma(G)$; here $|\Gamma(G)| = 2^{|E|}$.

A mixed graph approach for solving the problem $J \mid r_i \mid \sum w_i T_i$ is based on the following claims; Tanaev et al. (1994). A circuit-free directed graph $G_r = (Q, A \cup A_r, \varnothing) \in \Gamma(G)$ determines a semi-active schedule $S(G_r)$ for the problem $J \mid r_i \mid \sum w_i T_i$ and viceversa. There exists a one-to-one correspondence between the set of all the semi-active schedules and the set $\Gamma(G)$ of all the circuit-free directed graphs $G_r = (Q, A \cup A_r, \varnothing)$ generated by the mixed graph G via orienting edges of set E. Using a circuit-free directed graph $G_r \in \Gamma(G)$, the corresponding semi-active schedule $S(G_r)$ can be constructed via the critical path method in $O(|Q|^2)$ time. Thus, in terms of the mixed graph model $G = (Q, A, E)$, the most difficult question while solving the problems $J \mid r_i \mid \sum w_i T_i$ is to choose a circuit-free directed graph $G_r = (Q, A \cup A_r, \varnothing) \in \Gamma(G)$ that has a minimal value of the objective function (2) among those in all the circuit-free directed graphs $G_h \in \Gamma(G)$. To find an answer to this question is NP-hard problem. The circuit-free directed graph $G_r = (Q, A \cup A_r, \varnothing) \in \Gamma(G)$ with the minimal value of the corresponding objective function (2) (or the objective function (3) or (4), respectively) is called optimal directed graph for the problem $J \mid r_i \mid \sum w_i T_i$ (for the problem $J \mid r_i \mid \sum w_i C_i$ or the problem $J \mid r_i \mid C_{\max}$).

4. Evaluating of the dispatching rules

To evaluate the efficiency of the different dispatching rules, an optimal scheduler (like a B&B algorithm) for the problem $J \mid r_i \mid C_{\max}$, is used to solve instances with the restricted sizes $n \times m$ in order to obtain an exact solution (or an approximate solution) to the problem $J \mid r_i \mid C_{\max}$ in a resonable CPU-time. The information about orientations of the conflict edges is stored in Table 1 (edge $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}] \in E$ is called conflict if each of its orientation causes increasing of the completion time of either operation $o_{rp}^{\mu(rp)}$ or operation $o_{ab}^{\mu(ab)}$). The last column in Table 1 indicates that the optimal decision made by a scheduler to resolve a conflict edge $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}] \in E$ in the mixed graph $G = (Q, A, E)$ while the optimal (or the best constructed) digraph $G_s = (Q, A \cup A_s, \varnothing)$ was obtained. If an arc $(o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)})$ with $r < a$ was added to the digraph $G_s \in \Gamma(G)$ to resolve the conflict edge $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}] \in E$, then the last column in Table 1 was filled with number 1. On the other hand, if the symmetric arc $(o_{ab}^{\mu(ab)}, o_{rp}^{\mu(rp)})$ was added to the digraph $G_s \in \Gamma(G)$, then the last column in Table 1 was filled with number -1. For each conflict edge $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}]$ treated while branching in a B&B algorithm, the characteristics corresponding to the priority dispatching rules $X^t, t \in \{1, 2, \ldots, z\}$, for the operations $o_{rp}^{\mu(rp)}$ and $o_{ab}^{\mu(ab)}$ processed on the same machine $\mu(rp) = \mu(ab) = \mu_k \in M$ are calculated and stored in the corresponding cells of the row in Table 1.

Table 1
Conflict resolutions in the optimal digraphs $G_r \in \Gamma(G)$)

Conflict edges	X^1	X^2	$\ldots$	X^z	Optimal arcs
$[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}]$	$x_{rp,ab}^1$	$x_{rp,ab}^2$	$\ldots$	$x_{rp,ab}^z$	*either* 1 *or* -1
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$[o_{kl}^{\mu(kl)}, o_{cd}^{\mu(cd)}]$	$x_{kl,cd}^1$	$x_{kl,cd}^2$	$\ldots$	$x_{kl,cd}^z$	*either* 1 *or* -1
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$[o_{uv}^{\mu(uv)}, o_{ef}^{\mu(ef)}]$	$x_{uv,ef}^1$	$x_{uv,ef}^2$	$\ldots$	$x_{uv,ef}^z$	*either* 1 *or* -1

The characteristic $x_{rp,ab}^t$ of the conflict edge $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}]$ corresponding to the priority dispatching rule X^t is defined as the relative difference of the priorities π_{rp}^t and π_{ab}^t of the operations $o_{rp}^{\mu(rp)}$ and $o_{ab}^{\mu(ab)}$. As a priority dispatching rule X^t, let us consider the shortest completion time rule (SCT-rule) for operation $o_{rp}^{\mu(rp)}$ and operation $o_{ab}^{\mu(ab)}$ which are connected by the conflict edge $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}] \in E$. Let completion times of the operation $o_{rp}^{\mu(rp)}$ and operation $o_{ab}^{\mu(ab)}$ in the directed subgraph $(Q, A, \varnothing)$ of the mixed graph $G = (Q, A, E)$ be equal to 80 and 60, respectively. One can calculate the value of the characteristic $x_{rp,ab}^t$ as follows: $x_{rp,ab}^t = \dfrac{\pi_{rp} - \pi_{ab}}{\max\{\pi_{rp}, \pi_{ab}\}} = \dfrac{80 - 60}{80} = 0.25$. The sign of the value $x_{rp,ab}^t$ shows which of the operations $o_{rp}^{\mu(rp)}$ or $o_{ab}^{\mu(ab)}$ has a priority to be processed first on the corresponding machine μ_k with respect to SCT-rule X^t. The absolute value of $x_{rp,ab}^t$ shows how much this superiority of the operation with the larger priority is? In particular, the positive sign of the value $x_{rp,ab}^t$

indicates that operation $o_{ab}^{\mu(ab)}$ has to be processed before operation $o_{rp}^{\mu(rp)}$ respecting to SCT-rule. The absolute value of $x_{rp,ab}^{t} = 0.25$ belonging to the segment $[-1,1]$ shows how much this superiority of the operation $o_{ab}^{\mu(ab)}$ is respecting to the SCT-rule. After generating Table 1, the scheduler calculates worthiness of the dispatching rules included in the database by assigning weights to them. The weight w_t indicates the efficiency of the priority dispatching rule X^t in optimal scheduling for the problem $J \mid r_i \mid C_{max}$. To be more precise, the weight w_i for the dispatching rule is equal to the percentage of successful decisions made on the basis of the priority dispatching rule X^t. A successful decision happens when a dispatching rule implies the same orientation of the conflict edge in the set E as the B&B algorithm made when an optimal directed graph G_r was constructed for the problem $J \mid r_i \mid C_{\max}$. Therefore, the weight w_t of the priority dispatching rule X^t is defined as follows:

$$w_t = \frac{\textit{number of successful orientations of the conflict edges due to } X^t}{\textit{number of the conflict edges included to Table } 1}.$$

It should be noted that there are a lot of priority dispatching rules which are used in a variety of heuristic algorithms for scheduling jobs $J_i \in J$ in the job-shop; see Haupt (1989), Muth and Thompson (1963), Panwalkar and Iskander (1977) among others.

5. DR-algorithm

To solve the train scheduling problem, we need an algorithm that schedule operations Q sequentially, e.g., operation $o_{i,j+1}^{\mu(i,j+1)}$ must be considered after operation $o_{ij}^{\mu(ij)}$. This property of the desired algorithm will allow us to control a situation for each train in each railroad section. Note that some famous scheduling algorithms like a shifting bottleneck one (Adams et al., 1998) need a lot of CPU-time in the case when number m of the machines (railroad sections) is considerably large than number n of the jobs (trains). Moreover, each time it is desirable to know: Is there any train $\tau_i \in J$ which is waiting for the railroad section $\mu_k \in M$ occupied by another train $\tau_j \in J$ or not? If such a train τ_j exists, one can reduce a waiting time of the train τ_i by increasing the speed of the train τ_j crossing the railroad section $\mu_k \in M$. So, despite of the existence of many heuristic algorithms for the job-shop scheduling problems, we developed a new sequential algorithm named DR-algorithm to solve the problem $J \mid r_i \mid \sum w_i T_i$, which is more appropriate for the train scheduling. The DR-algorithm generates a sequence of the operations $o_{ij}^{\mu(ij)} \in Q$ processed on different machines of the set M in the order such that they are requested for processing the jobs $\tau_i \in J$. During the first iteration, the DR-algorithm finds the first request (i.e., operation $o_{i,1}^{\mu(i,1)}$) of a job $\tau_i \in J$. The operation $o_{i,1}^{\mu(i,1)}$ is compared with all other operations processed on the same machine $\mu(i,1) = \mu_k \in M$. In the scheduling process for each conflict edge $[o_{ij}^{\mu(ij)}, o_{gh}^{\mu(gh)}] \in E$ of the mixed graph $G = (Q, A, E)$, which was met by the scheduler, the characteristic vector $(x_{ij,gh}^{1}, x_{ij,gh}^{2}, \ldots, x_{ij,gh}^{z})$ is calculated. The priority value $pv_{ij,gh}$ is calculated as follows:

$$pv_{ij,gh} = \sum_{t=1}^{z} (w_t \times x_{ij,gh}^{t}).$$

If the priority value $pv_{ij,gh}$ is positive and $i < g$, then the arc from vertex $o_{ij}^{\mu(ij)}$ to vertex $o_{gh}^{\mu(gh)}$ is added to the resulting digraph G_r. If the value $pv_{ij,gh}$ is negative, then the symmetric $(o_{gh}^{\mu(gh)}, o_{ij}^{\mu(ij)})$ is added

to the digraph G_r. For example, at the first iteration of the algorithm, the DR-algorithm compares operation $o_{i,1}^{\mu(i,1)}$ with all operations $o_{jk}^{\mu(jk)}$ of the other jobs $\tau_j \in J$ on the same machine $\mu(i1) = \mu_d \in M$ processing operations $o_{jk}^{\mu(jk)}$, $\tau_j \in J$. If the priority value $pv_{ij,gh}$ is positive, then an arc starting from the vertex $o_{i1}^{\mu(i1)}$ and ending to the vertex $o_{jk}^{\mu(jk)}$ has to be added to the desired digraph G_r. Otherwise, the symmetric arc $(o_{jk}^{\mu(jk)}, o_{i1}^{\mu(i1)})$ with $j > i$ is added to the digraph G_r.
After sequencing operations $o_{i1}^{\mu(i1)}$ for all jobs $\tau_i \in J$, the DR-algorithm considers operation $o_{i2}^{\mu(i2)}$ for each job $\tau_i \in J$, then operations $o_{i3}^{\mu(i3)}$ for each job $\tau_i \in J$, and so on until operation $o_{in_i}^{\mu(in_i)}$ for each job $\tau_i \in J$ being considered.

6. Train tardiness reduction via controllable scheduling

A three stage scheduling algorithm was used to reduce the total job tardiness (or delay time of the trains). In the first stage, the trains J are pre-scheduled by the DR-algorithm. In the second stage, the tardiness of each train is measured and the algorithm tries to modify the departure time of the trains to decrease the total tardiness on the basis of information obtained at the pre-scheduling stage. In the third stage, the special module is used to improve the quality of the preliminary schedule and to construct the final schedule (see Fig 1).

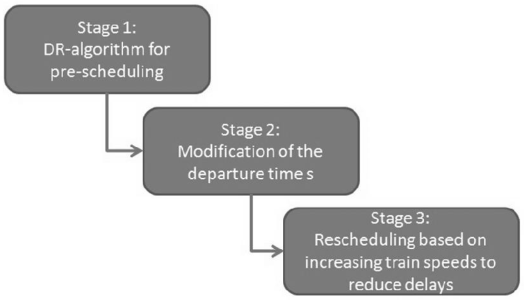
Stage 1:
DR-algorithm for pre-scheduling
Stage 2:
Modification of the departure time s
Stage 3:
Rescheduling based on increasing train speeds to reduce delays

Fig. 1. Three stage scheduling

6.1. Modifying the departure times of the trains

The idea behind modifying a train departure time is that due to changing the departure time d_t of the train $\tau_t \in J$ by departure time d_t' belonging to the permitted interval, $-\alpha \le d_t' \le +\alpha$, one can reduce the number of conflicts between trains tending to use the same railroad section at the same time. In this stage of train scheduling, the tardiness T_j (see Eq. (2)) for each train $\tau_j \in J$ is calculated. The average tardiness A_{ve} of the n trains is calculated as $A_{ve} = \frac{\sum_{j=1}^{n} T_j}{n}$ and set J of all the trains is divided into three subsets as follows:

- $T_j < A_{ve} - \beta$: If tardiness T_j of the train τ_j is smaller than the average value A_{ve}, it shows that the other trains of the set J wait more (in average) for the train τ_j than train τ_j waits for other trains from the set J. Therefore, if the train τ_j will start earlier, it may release the railroad sections earlier and this could reduce the tardiness of other trains.
- $T_j > A_{ve} + \beta$: If tardiness T_j of the train τ_j is larger than the average value A_{ve}, it shows that train τ_j waits for other trains more than other trains wait for the train τ_j. In such a situation, if the train τ_j will departs later, it may have less conflicts with other trains for the railroad sections.

- $A_{ve} - \beta \le T_j \le A_{ve} + \beta$: The tardiness of the train τ_j belongs to the feasible range of the average tardiness A_{ve}. No change for the departure time r_i is needed. The value β shows the range of the tardiness connivance. Finding a proper value for the β value depends on the concrete train scheduling problem and so the value of β is defined by the user.

The above procedure is used to modify the start times of the trains (the release times of the jobs) J. Maximal possible change α of the departure time ($-\alpha \le d_t \le +\alpha$) must be assigned by the user. In our computational experiments, the value α was equal to $\pm 10\%$ of A_{ve}.

6.2. Train speed control

In the third stage, DR-algorithm reschedules the trains J on the railroad sections M. After resolving conflict edges $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}] \in E$, a speed control module is applied. The speed control module compares completion time c_{ij} of the operation $o_{ij}^{\mu(ij)}$ on machine $\mu_d = \mu(ij)$ with the minimal release times r_{kl} of other operations $o_{kl}^{\mu(kl)}$ with $\mu(kl) = \mu_d \in M$. Three situations have to be considered:

- If $c_{ij} \le r_{kl}$, then there is no a competition between two trains τ_i and τ_k to use the railroad section $\mu_d = \mu(ij) = \mu(kl)$. Therefore, the scheduling process is continued without changing the speed of train τ_i.
- If both inequalities $c_{ij} > r_{kl}$ and $c_{ij} - r_{kl} > p_{ij} \times \Omega\%$ hold, then increasing the speed of train τ_i may decrease the tardiness of train τ_k. Therefore, the scheduler increases the speed of train τ_i by $\Omega\%$. As a result, train τ_i will release the railroad section $\mu_d = \mu(ij) = \mu(kl)$ earlier. Let Ω be equal to 10%, then $p_{ij} = p_{ij} - (p_{ij} \times 10\%)$.
- If both inequalities $c_{ij} > r_{kl}$ and $c_{ij} - r_{kl} \le p_{ij} \times \Omega\%$ hold, then increasing the speed of the train τ_i is desirable, however this increasing may be no more than $\Omega\%$. So, the time used by train τ_i to cross the railroad section $\mu(ij) = \mu(kl) = \mu_d \in M$ will be decreased by the value $c_{ij} - r_{kl}$ and so $p_{ij} = p_{ij} - (c_{ij} - r_{kl})$.

This procedure allows the scheduler to increase the speed of trains with a feasible $\Omega\%$ in order to reduce the train tardiness (if any). Of course, there is a limitation on such a speed increase depending on the train types, railway, environmental situation, etc.

6.3. Example

The following example allow us to demonstrate the main idea of the proposed scheduling algorithm. We assume that three trains $\{\tau_1, \tau_2, \tau_3\} = J$ have to be scheduled on three railroad sections $\{\mu_1, \mu_2, \mu_3\} = M$. The operation set $Q = \bigcup_{k=1}^{m} Q^k$ (where $m = 3$) includes the following three subsets: $Q^1 = \{o_{1,1}^1, o_{2,2}^1, o_{3,3}^1\}$, $Q^2 = \{o_{1,2}^2, o_{2,3}^2, o_{3,1}^2\}$, and $Q^3 = \{o_{1,3}^3, o_{2,1}^3, o_{3,2}^3\}$ (see Fig 2). The departure (release) times for trains (jobs) $\tau_i \in J$ are given as $r_1 = 22$, $r_2 = 3$, and $r_3 = 14$. At the first stage, pre-scheduling was executed. The total tardiness for all three trains after pre-scheduling is equal to 43 ($\sum_{i=1}^{3} T_i == 0 + 23 + 20 = 43$), the total completion time is equal to 426 ($\sum_{i=1}^{3} C_i = 139 + 140 + 147 = 426$), and the makespan is equal to 147 ($\max_{i=1}^{3} C_i = \max\{139, 140, 147\} = 147$) (see Fig 3).

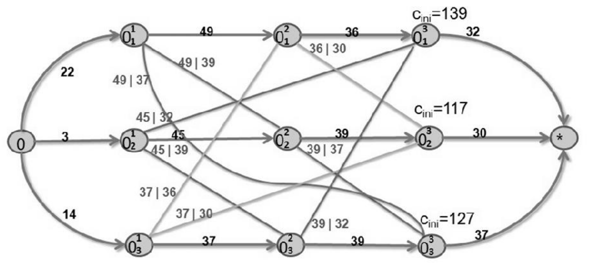

Fig. 2. Mixed graph $G = (Q, A, E)$ for three trains that must pass three railroads sections

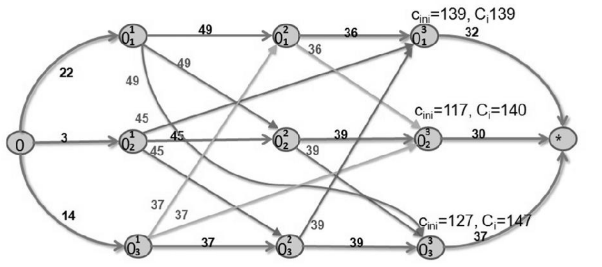

Fig. 3. Directed graph $G_r \in \Gamma(G)$ constructed by DR-algorithm at the first stage

At the second stage, after comparing the train delays with average delay time, the new departure (release) times were assigned to the trains $\tau_i \in J$ as follows: $r_1 = 15$, $r_2 = 10$, and $r_3 = 7$ (see Fig 4). The total tardiness for all trains now is equal to 34 $(\sum_{i=1}^{3} T_i == 0 + 9 + 25 = 34)$, the total completion time is equal to 405 $(\sum_{i=1}^{3} C_i = 132 + 133 + 140 = 405)$, the makespan is equal to 140 $(\max_{i=1}^{3} C_i = \max\{132, 133, 140\} = 140)$.

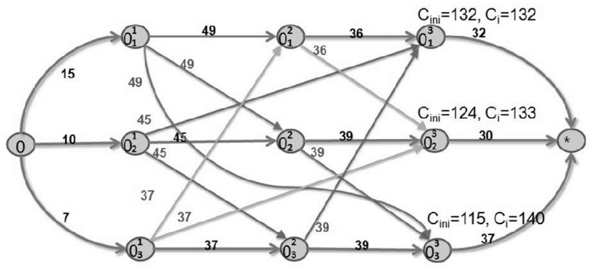

Fig. 4. The modified departure (release) times for trains (jobs) calculated at the second stage

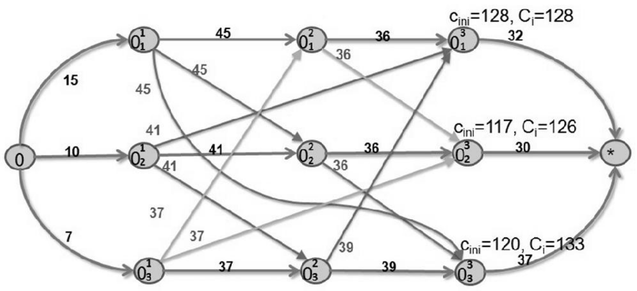

Fig. 5. Directed graph G_s obtained from the mixed graph G due to rescheduling based on the modified train speeds and departure times (in the third stage)

At the third stage, the train speeds are modified (as it is explained in Subsection 6.2), and the jobs (trains) J are rescheduled again via resolving conflict edges $[o_{rp}^{\mu(rp)}, o_{ab}^{\mu(ab)}] \in E$ provided that train speeds are modified. By comparing Fig 2 with Fig 5, it can be seen that three processing time are changed and the summation of reduction is equal to eleven time unit. As a result due to reducing the given processing times p_{ij} by at most 10%, the total tardiness is reduced to 22 time units $(\sum_{i=1}^{3} T_i = 0 + 9 + 13 = 22)$, the total completion time is reduced to 387 time unites $(\sum_{i=1}^{3} C_i = 128 + 126 + 133 = 387)$, and the makespan is reduced to 133 $(\max_{i=1}^{3} C_i = \max\{128, 126, 133\} = 133)$ time units (see Fig 5).

7. Computational results

DR-algorithm was coded in Borland Delphi. For evaluating the efficiency of the developed algorithm, we compared it with the results of the six heuristic dispatching rules, which were also coded in Borland Delphi. These heuristic algorithms are based on the following priority dispatching rules: Shortest Release Time rule (Algorithm SReT), Shortest Start Time rule (Algorithm SStT), Longest Delay rule (Algorithm LDelay), Shortest Completion Time rule (Algorithm SCT), Earliest Due Date rule (Algorithm DueDate), and Smallest Number of Remaining Jobs rule (Algorithm SNJR). The DR-algorithm was compared with these six heuristic algorithms for the makespan criterion $C_{\max}$ (Table 2), for the total tardiness criterion $\sum T_i$ (Table 3), and for the total completion time criterion $\sum C_i$ (Table 4). In the experiments, we used 20 benchmark job-shop instances introduced by Lawrence (1984) (i.e. instances la01 – la20) to evaluate the seven developed heuristic algorithms. The minimal possible makespans for the instances la01 – la20 are known from Internet and they are given in the last column of Table 2. In column 1 of Tables 2 – 4, the names of the benchmark instances are given, in column 2 the sizes $n \times m$ of the problems, in columns 3 - 7 the objective values obtained by the corresponding heuristic algorithms.

Table 2
A comparison of DR-algorithm and heuristic scheduling rules for the makespan criterion

Job-shop	Size	SReT	SStT	LDelay	SCT	DueDate	SNJR	DR-algorithm	Optimal
la01	10 × 5	728	1080	1867	803	1135	749	774	666
la02	10 × 5	908	858	1724	849	1030	924	706	655
la03	10 × 5	795	1080	1313	794	900	861	690	597
la04	10 × 5	749	948	1839	833	1254	828	766	590
la05	10 × 5	819	838	1510	939	782	643	$\underline{593}$	593
la06	15 × 5	1371	1175	2241	1500	1293	1047	$\underline{926}$	926
la07	15 × 5	1243	1153	2017	1114	1393	1037	973	890
la08	15 × 5	1186	1175	1941	1126	1453	1079	935	863
la09	15 × 5	1113	1182	2241	1342	1210	1010	$\underline{951}$	951
la10	15 × 5	1094	1197	1935	1610	1392	1074	$\underline{958}$	958
la11	20 × 5	1372	1567	2695	1518	1844	1282	$\underline{1222}$	1222
la12	20 × 5	1313	1364	2136	1732	1859	1231	$\underline{1039}$	1039
la13	20 × 5	1596	1427	2561	1772	2076	1189	1176	1150
la14	20 × 5	1426	1439	2734	1493	1675	1292	$\underline{1292}$	1292
la15	20 × 5	1687	1816	2718	1776	2337	1569	1294	1207
la16	10 × 10	1254	1323	3536	1183	1422	1191	1105	945
la17	10 × 10	962	1396	2597	1086	1151	959	813	784
la18	10 × 10	1218	1181	3846	1207	1374	1111	976	848
la19	10 × 10	1167	1094	2983	1204	1335	1087	936	842
la20	10 × 10	1273	1177	3699	1414	1422	1087	980	902

In Tables 2 – 4, the best objective values obtained by the seven heuristic algorithms are presented in boldface. In Table 2, the optimal makespan values obtained by DR-algorithm are underlined. Other six heuristic algorithm did not obtain the minimal makespan values.

Table 3
A comparison of DR-algorithm with six heuristic algorithms based on despatching rules for the criterion $\sum T_i$

Job-shop	Size	SReT	SStT	LDelay	SCT	DueDate	SNJR	DR-algorithm
la01	10 × 5	3237	3660	9924	3040	7314	3744	3769
la02	10 × 5	4415	3731	8692	3752	6062	4261	3072
la03	10 × 5	4070	4708	5937	3146	5588	4681	3860
la04	10 × 5	3170	4064	9184	3472	8464	4633	4239
la05	10 × 5	3669	3198	8365	4010	4497	3485	2837
la06	15 × 5	10750	8805	19252	10010	13454	9769	7778
la07	15 × 5	10398	9012	16184	7268	14441	9592	8814
la08	15 × 5	9152	9198	15632	7109	15329	9960	8239
la09	15 × 5	8520	8771	19043	9802	11526	8910	8274
la10	15 × 5	8115	9517	16178	10006	13158	9528	8941
la11	20 × 5	16711	17654	29100	16826	27715	16597	15681
la12	20 × 5	13778	14943	24532	15652	28209	15363	14234
la13	20 × 5	17905	14789	31431	17561	32150	16210	16428
la14	20 × 5	15824	16846	31509	15163	24369	17287	17822
la15	20 × 5	19486	21594	30930	19305	35743	18764	15132
la16	10 × 10	4505	5212	18449	3364	7636	5001	4437
la17	10 × 10	3177	5709	12219	4005	6117	4239	3004
la18	10 × 10	4088	4511	18410	4290	7496	4531	3839
la19	10 × 10	4489	3839	13718	3510	7321	4582	3164
la20	10 × 10	4835	3594	18082	4504	8131	4502	3579

From Table 2, it follows that the DR-algorithm is more effective for the objective of minimizing the makespan (C_{max} criterion) than six heuristic algorithms based on the pure dispatching rules. DR-algorithm obtained the smaller makespans for all solved instance la01 – la20 with only two exceptions (instances la01 and la04). Optimal makespan values were obtained by DR-algorithm for seven treated instances. For the total tardiness objective ($\sum T_i$ criterion) the DR-algorithm has superiority comparing to other six heuristic algorithms. The DR-algorithm was eleven times in the first place among ather algorithms tested (see Table 3). In Table 4, the DR-algorithm is compared with six heuristic algorithms for minimizing the maximal job completion time (criterion $\sum C_i$). The DR-algorithm is the best one for solving thirteen benchmark instances.

Table 4

A comparison of DR-algorithm and six heuristic algorithms base on dispatching rules for the criterion $\sum C_i$

Job-shop	Size	SReT	SStT	LDelay	SCT	DueDate	SNJR	DR-algorithm
la01	10 × 5	4760	5183	11447	4563	8837	5267	5292
la02	10 × 5	5776	5092	10053	5113	7423	5622	4433
la03	10 × 5	5737	6375	7604	4813	7255	6348	5527
la04	10 × 5	4643	5537	10657	4945	9937	6106	5712
la05	10 × 5	5158	4686	9854	5499	5986	3485	4974
la06	15 × 5	13174	11229	21676	12434	15878	12193	10202
la07	15 × 5	12721	11335	18507	9591	16764	11915	11137
la08	15 × 5	11707	11753	18187	9664	17884	12515	10794
la09	15 × 5	10963	11214	21486	12245	13969	11353	10717
la10	15 × 5	10413	11815	18476	12304	13158	11826	11239
la11	20 × 5	19994	20973	32383	20104	30998	19880	18964
la12	20 × 5	16476	17641	27230	18350	30907	18061	16932
la13	20 × 5	21547	18431	35073	21203	35792	19852	19070
la14	20 × 5	19094	20116	34779	18433	27639	20557	21092
la15	20 × 5	23069	25177	34513	22888	39326	22347	18715
la16	10 × 10	8526	9233	22470	7358	11657	9022	8458
la17	10 × 10	7015	9547	16057	7843	9955	8077	6842
la18	10 × 10	8310	8733	23632	8512	11718	8753	8061
la19	10 × 10	8751	8101	17980	7772	11583	8844	7426
la20	10 × 10	9352	8111	22599	9021	12648	9019	8096

For all three objective functions, which are considered in train scheduling, the developed DR-algorithm was better than six heuristic algorithms tested. In order to evaluate the DR-algorithm on reducing the total completion time, makespan and total tardiness objective in our three stage strategy, the different job-shop problems (with sizes from 3×3 to 10×10) have been generated. The times p_{ij} to pass a railroad section $\mu(ij) \in M$ by train $\tau_i \in J$ are randomly generated in the segment [30,50]. The maximal possible reduction of the train speed was restricted by 10% of the original time p_{ij} (i.e., the maximal possible reduction is equal at most to 5 time units).

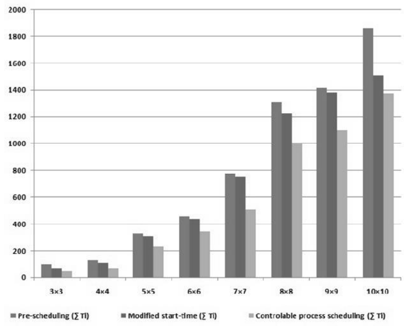

Fig. 6. The total tardiness of the trains after each of three stages.

Fig. 6 presents the total tardiness of the trains obtained after the first stage, the second stage, and third stage 3 of the DR-algorithm. By an overview given in Fig 6, it is observable that the total tardiness in average has been reduced about 20%.
In Table 5, the total completion time and makespan objective function for DR-algorithm are presented after each scheduling step (columns 2 – 4 for criterion $\sum C_i$ and columns 5 – 7 for criterion C_{max}). Column 8 in Table 5 presents the number of modified operations $o_{ij}^{\mu(ij)}$, i.e., the number of railroad sections were a train speed was increased. Column 9 presents the total volume of changes of the processing times p_{ij} due to increasing the speeds of some trains.

Table 5
Running DR-algorithm with modified train speeds.

Job-shop size	$\sum C_i$ stage 1	stage 2	stage 3	C_{max} stage 1	stage 2	stage 3	Number of modified operations	The total change of p_{ij}
3×3	510	495	475	209	204	196	2	8
4×4	812	772	733	261	251	240	6	20
5×5	1343	1308	1238	312	302	289	15	50
6×6	1977	1953	1864	410	406	388	13	46
7×7	2848	2782	2538	513	503	438	29	96
8×8	3997	3894	3666	591	634	604	35	117
9×9	4690	4677	4396	587	586	616	41	133
10×10	5880	5510	5377	751	683	654	52	172

8. Conclusion

A three-stage strategy was used to reduce the delays in train scheduling. In the first stage, a pre-scheduling algorithm executed to achieve some data about trains. At the second stage, the departure time of the trains are modified in order to decrease train delays. In the third stage, a controllable processing time module tries to reduce train delays as much as possible via increasing speeds of some trains. Computational results shows that this policy can be useful to reduce total tardiness, total completion time and makespan in the train scheduling. As a future research, it is desirable to consider priorities for usual trains and non-stop type of trains. For defining the right weights for the dispatching rules, we used some optimal solutions available for the benchmark problems la01 – la20 with the makespan objective function in order to assign appropriate weights to the dispatching rules. However, these weights may not be so useful for the criteria $\sum T_i$ and $\sum C_i$ as for the makespan criterion. Therefore, the computational results for these two objective functions (Tables 3 and 4) are not so impressive like those obtained for the makespan criterion (Table 2). Therefore, it will be useful to develop exact algorithms for the problems $J \mid r_i \mid \sum T_i$ and $J \mid r_i \mid \sum w_i C_i$ with objectives $\sum T_i$ and $\sum C_i$.

References

Abdolzadeh, M., & Rashidi, H. (2010). An approach of cellular learning automata to job shop scheduling problem. *International Journal of Simulation: Systems, Science and Technology*, 34, 391-401.

Adams J., Balas, E., & Zawack, D. (1998). The shifting bottleneck procedure for job-shop scheduling. *Management Science*, 11, 2, 56-64.

Burdett, B., & Kozan, E. (2010). A disjunctive graph model and framework for constructing new train schedules, *European Journal of Operational Research*. 200, 85–98.

Cai, X., & Goh, C.J. (1994). A fast heuristic for the train scheduling problem, *Computers & Operations Research*. 21, 499–510.

Carey, M., & Lockwood, D. (1995). A model, algorithms and strategy for train pathing. *Journal of*

Operational Research Society, 46, 8, 988-1005.

Dorfman, M.J., & Medanic, J. (2004). Scheduling trains on a railway network using a discrete event model of railway traffic, *Transportation Research, Part B,* 38, 81–98.

Dorndorf, U., & Pesch, E. (1995). Evaluation based learning in a job shop scheduling environment. *Computers & Operations Research*, 22, 1, 25-40.

Gabel, T., & Riedmiller, M. (2007). Adaptive reactive job-shop scheduling with learning agents. *International Journal of Information Technology and Intelligent Computing, IEEE Press*, 2, 4.

Geiger, C.D., Uzsoy, R., & Aytug, H. (2006). Rapid modelling and discovery of priority dispatching rules: an autonomous learning approach. *Journal of Scheduling*, 9, 7-34.

Ghoseiri, K., Szidarovsky, F., & Asgharpour M. (2004). A multi-objective train scheduling model and solution, *Transportation Research Part B: Methodological,* 38, 927–952.

Glover, F. (1989). Tabu search – part 1. *ORSA Journal on Computing.* 1, 190-206.

Graham, R.L., Lawler, E.R., Lenstra, J.K., & Rinnooy Kan, A.H.G. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5, 287-326.

Haupt, R. (1989). A survey of priority rule-base scheduling. *OR Spectrum*, 11, 1, 3-16.

Jovanovic, D., & Harker, P.T. (1991). Tactical scheduling of rail operations: the scan i system. *Transportation Science*, 25, 1, 46-64.

Kraay, D., Harker, P.T., & Chen, B. (1991). Optimal pacing of trains in freight railroads: model formulation and solution. *Operaions Research*, 39, 1, 82-99.

Lawrence, S. (1984). Supplement to resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques, *Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh*, *USA*.

Li, D.C., & Shi, I.S. (1994). Using unsupervised learning technologies to induce scheduling knowledge for FMSs. *International Journal of Production Research*, 32, 9, 2187-2199.

Lusby, R., Larsen, J., Ehrgott, M., & Ryan, D. (2011). Railway track allocation: models and methods. *Operations Research Spektrum*, 33, 843-883.

Mladenovic, S., & Cangalovic, M. (2007). Heuristic approach to train rescheduling. *Yugoslav Journal of Operations Research*, 17, 1, 9-29.

Muth, J.F. Thompson, G.L. (1963). *Industrial Scheduling.* Prentice-Hall, Englewood Cliffs, N.J.

Naderi-Beni, M., Tavakkoli-Moghaddam, R., Naderi, B., Ghobadian, E., & Pourrousta, A. (2012). A two-phase fuzzy programming model for a complex bi-objective no-wait flow shop scheduling, *International Journal of Industrial Engineering Computations,* 3, 617–626

Pacciarelli, D., & Pranzo, M. (2001). A tabu search algorithm for the railway scheduling problem, *Proceedings of the 4th Meta-heuristics International Conference, MIC'2001,* 159–165.

Panwalkar, S.S., & Iskander, W. (1977). A survey of scheduling rules. *Operations Research*, 25, 1, 45-61.

Shafia, M. A., Pourseyed Aghaee, M., & Jamili, A. (2010). A new mathematical model for the job shop scheduling problem with uncertain processing times. *International Journal of Industrial Engineering Computations,* 2, 295-306.

Szpigel, B.(1973). Optimal train scheduling on a single line railway. *Operaions Research*, 72, 344-351.

Tanaev, V.S., Sotskov, Y.N., & Strusevich, V.A. (1994). *Scheduling Theory: Multi-Stage Systems.* Kluwer Academic Publishers, Dordrecht, The Netherlands.

Thulasiraman, K., & Swamy, M.N.S. (1992). *Graph: Theory and Algorithms.* John Wiley & Sons, Inc., New York, USA.

Van Laarhoven, P.J.M., Aarts, E.H.L., & Lenstra, J.K. (1992). Job shop scheduling by simulated annealing. *Operations Research.* 40, 113–125.

Zhou, X., & Zhong, M. (2007). Single-track train timetabling with guaranteed optimality: branch-and-bound algorithms with enhanced lower bounds. *Transportation Research Part B*, 21, 320-341.