A comparison of approaches to modeling train scheduling problems as job-shops with blocking constraints

Julia Lange Frank Werner

Institute of Mathematical Optimization
Otto-von-Guericke-University Magdeburg

julia.lange@ovgu.de · frank.werner@ovgu.de

October 28, 2015

Abstract

A train scheduling problem is considered, where a set of trains travels through a given railway network consisting of single tracks, sidings and stations. For every train a fixed route and travel times, an earliest departure time at the origin and a desired arrival time at the destination are given. To increase customer satisfaction and planning certainty in adjacent networks, the goal is to avoid trains arriving later than the desired time. The train scheduling problem is interpreted as a job-shop scheduling problem, where jobs represent trains and machines constitute tracks or track sections. For every job a technological order, in which its operations are to be processed, is given by the train route. The release times and due times of the jobs as well as the processing times for all operations are known by the input data of the problem. A feasible schedule minimizing the total tardiness of all jobs is to be determined. Blocking constraints are additionally included to account for a train blocking a track until the succeeding track is free to travel on. Dependent on the transformation approach applied a job-shop scheduling problem with or without additional routing flexibility in stations and sidings is to be solved. For this NP-complete problem several mixed integer programming formulations based on different transformation approaches using distinct types of decision variables are derived and compared in terms of total tardiness values, formulation size and computation time.

Keywords: Scheduling · Job-Shop · Total Tardiness · Blocking · Mixed Integer Programming

MSC: 90B35 · 90C11 · 90C27

1 Introduction

Train scheduling problems have been studied with increasing intensity in different research fields. The main motivation arises out of the necessity to schedule trains passing through networks of increasing size and complexity in real-world situations. Since rail vehicles are well established

and in many countries the only means of public and freight transportation, the demand for an efficient use of railway networks and for smart scheduling techniques remains important in the future.

A train scheduling problem can be described as follows. A set of trains traveling through a predefined network with different routes and entering times is given. The network consists of track sections, which contain either a single track or several parallel tracks. The order of the trains is to be determined for every track. Following a regular optimization criterion, the starting times of the trains on the tracks and with it a schedule is automatically defined by the order.

In many papers the time frame between the starting time of the first and the arrival time of the last train is minimized. This criterion is not reasonable for passenger and freight transportation, since it does not account for a balanced distribution of waiting times. Some trains may have long waiting times while others have none. In order to increase customer satisfaction the minimization of the total travel time of all trains is convenient. With additional regard to the planning certainty in adjacent railway networks, the *minimization of the sum of tardiness times* of all trains according to desired due times is the optimization criterion considered here.

A specific characteristic of train scheduling problems compared to other scheduling problems is the appearance of *blocking constraints*. These constraints refer to situations in which a train occupies a track section longer than necessary until the succeeding section is free to travel on. The effect of these blocking restrictions on feasibility, optimal solutions and computation times will be examined in detail in the following. Several further conditions in train scheduling, such as deadlocks and train length (see e.g. Liu and Kozan (2009)), headways (see e.g. D'Ariano et al. (2007)), acceleration and deceleration (see e.g. Burdett and Kozan (2010)) and meetings of trains (see e.g. Oliveira and Smith (2000)) can be included in the problem to match real-world situations, however will not be regarded here.

This paper will focus on *single-track networks*, which mainly consist of bidirectional single tracks connecting stations. Each single track can be occupied at most by one train at a time, so that the only possibility for trains to overtake is given in sidings and stations with different numbers of parallel tracks.

Even a simple form of a single-track train scheduling problem is classified to be NP-complete in Cai and Goh (1994). Combined with the ongoing necessity of solving real-world train scheduling problems, this seems to be a reason for many efforts being made towards designing and adapting procedures to generate feasible, potentially near-optimal solutions. The variety of approaches implemented to describe, model and solve train scheduling problems includes MIP and IP formulations with different assignment structures (see e.g. Jovanovic and Harker (1991), Carey and Lockwood (1995) and Lamorgese and Mannino (2013)), graph-based methods such as disjunctive programming, graph coloring and set-packing and -partitioning formulations (see e.g. Burdett and Kozan (2010) and Gholami et al. (2013)), meta-heuristic methods such as tabu search, simulated annealing and evolutionary algorithms (see e.g. Van Laarhoven et al. (1992), Higgins et al. (1997) and Gröflin and Klinkert (2009)), as well as specialized and extended branch and bound procedures (see e.g. Sauder and Westerman (1983) and Cacchiani et al. (2013)). A broad overview of different problem structures and solution approaches is given in Cordeau et al. (1998)

and Lusby et al. (2011).

In the following the train scheduling problem will be regarded as a *job-shop scheduling problem*. Trains are considered as jobs processed on machines which in turn represent tracks or track sections. The train routes define the technological orders of the jobs. Special attention is paid to the transformation of stations and sidings with parallel tracks. Referring to the literature, sections with parallel tracks can be interpreted as parallel machines, one machine with parallel machine units or intermediate buffers with a capacity equal to the number of parallel tracks. The first two adaptions, namely the *Parallel-Machine Approach* and the *Machine-Unit Approach*, are compared in a computational study here.

Based on these transformations, mixed integer programming formulations with different assignment variables are set up to determine an optimal schedule. The assignment variables basically refer to the disjunctive graph representation of scheduling problems with which feasible order relations between the operations requiring the same machine can be determined. The *precedence variables* define the precedence relation between all pairs of operations having to be processed on the same machine directly. In contrast, the *order variables* assign a unique order position to each operation on a machine and define precedence relations between pairs indirectly.

The effects of applying different transformation approaches and assignment variables on the computation time and the objective function value will be investigated in the following. Randomly generated instances with up to 20 jobs will be tested in a computational study using IBM ILOG CPLEX 12.6.1. The underlying idea and some of the results have already been addressed in Lange (2015). This paper includes more detailed explanations on transformation methods and decision variables. Additionally, computational results of larger test instances and extended comparisons of different MIP formulations are given.

The paper is organized as follows. Section 2 gives a detailed description of the train scheduling problem with special attention given to underlying restrictions and assumptions. Section 3 summarizes the transformation of the problem to a job-shop problem with blocking constraints. In Section 4 an overview of the modeling approaches is given, where the representation of parallel tracks (4.1) and the differentiation of assignment variables (4.2) are addressed separately.

The following Sections 5 and 6 present MIP formulations based on the Parallel-Machine Approach and on the Machine-Unit Approach, each starting with basic remarks on the notation and the input data in 5.1 and 6.1, respectively. For each of both modeling approaches the resulting optimization programs are given and explained applying precedence variables on the one hand (Sections 5.2 and 6.2) and order variables on the other (Sections 5.3 and 6.3). In Section 7 the computational study comparing the four MIP formulations is presented by explaining the generation of the test instances in 7.1 and analyzing the computational results in 7.2. Section 8 gives a short overview of potential extensions to the problem. To conclude, the main results are summarized in Section 9 together with some remarks on future research perspectives.

2 The train scheduling problem

Given is a railway network consisting of bidirectional single tracks connecting stations with different quantities of parallel tracks. Since a track can be occupied only by one train at a time, it is only possible for trains to overtake at stations and sidings. A siding is a small parallel track for trains to wait and overtake lacking a platform for passengers to get on or off.
To structure and simplify the scheduling process, the railway network is split up into track sections formed by single tracks, sidings and stations, where each train can only occupy one track section at a time. In contrast, each track section can be occupied by as many trains as it has parallel tracks. The capacity of a single track is one by definition, whereas sidings and stations may consist of different numbers of parallel tracks. An example of a single-track network is given in Figure 1. The network consists of four bounding nodes A, B, D and E, which are assumed to be stations with infinite capacity. C is a station with two parallel tracks and the single track between station C and the branch D,E has a siding.

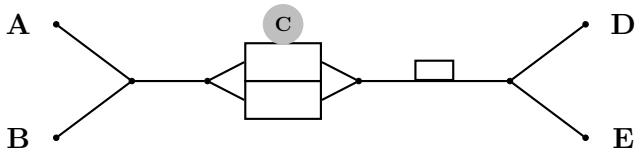

Figure 1: Single-track network

Since the trackage regarded constitutes a fraction of an underlying national or international railway network, a train enters the considered network at a certain moment and is supposed to leave it at another point in time. The route through the network is predefined for each train and it is assumed that routes can only begin and end at bounding nodes. Thus the problem can be stated as scheduling all trains on the track sections according to their routes, so that the desired leaving times are reached.
A special characteristic of trains as items to be scheduled is that they cannot easily be stored aside after passing a track section. This means that a train blocks a track after passing it until the succeeding section on its route is free to travel on. This situation is referred to as blocking.
Another important aspect to take into account for reasons of safety is the implementation of headways, which define a minimum distance between trains traveling in the network. Since the underlying assumptions are that all tracks are sufficiently large to capture the trains and that only one train is allowed to travel on a track at a time, headways are implicitly given and will not be regarded as additional constraints.

3 The job-shop scheduling problem

The idea of interpreting a train scheduling problem as a job-shop scheduling problem was initially published in Szpigel (1973) and is pursued by many other authors such as in Kreuger et al. (1997), Oliveira and Smith (2000), D'Ariano et al. (2007), Liu and Kozan (2009) and Gholami et al. (2013). In accordance to the literature, the underlying transformation is done as described below. An overview of the notation applied is given in the appendix.

Given is a single-track network with tracks/track sections defining the set of machines

$$\mathcal{M} = \{M_k \mid k = 1, \ldots, m\}.$$

The trains traveling in the network are described by the set of jobs

$$\mathcal{J} = \{J_i \mid i = 1, \ldots, n\}.$$

An operation is defined by a machine processing a job without interruption, which corresponds to a train passing through a track section without stopping. Following the route of each train each job has a certain technological order, according to which it is to be processed on different machines. An ordered set of operations $\mathcal{O}^i = \{O_{ij} \mid j = 1, \ldots, n_i\}$ is given for each job J_i. In order to allow periodic and cyclic train routes, recirculation (recrc) of jobs is included, so that a job can be processed more than once on a machine. The set of operations is defined by $\mathcal{O} = \bigcup_{J_i \in \mathcal{J}} \mathcal{O}^i$. There exist precedence relations between operations of the same job implementing its technological order. The processing times p_{ij} of the operations correspond to the travel times of trains on track sections. Additionally, the release times r_i and the due times d_i of the jobs, which describe entering times and desired leaving times of the trains in the network, are given. A schedule is defined by the determination of the order of the jobs on the machines and with it the starting times s_{ij} for all operations. Using these starting times, the completion time C_i and the tardiness T_i of each job J_i can be calculated as follows:

$$C_i = s_{in_i} + p_{in_i} \quad \text{and} \quad T_i = \max\{0, C_i - d_i\}.$$

This paper will focus on the minimization of the total tardiness:

$$\sum_{J_i \in \mathcal{J}} T_i \longrightarrow \min!$$

The characteristic blocking situation occurring in train scheduling is implemented as jobs blocking machines until succeeding machines are idle. This restriction (block) is also known as a no-store constraint as addressed in D'Ariano et al. (2007), due to blocking being caused by the absence of a possibility to store the jobs (trains) elsewhere until processing can be continued on the next machine.

Since the direction of travel of the trains is not considered in the transformation, so called deadlock situations might come to pass when implementing a solution. Let J_i be processed on M_k and proceed to M_{k+1}, while J_{i+1} is processed on M_{k+1} and will proceed to M_k. In job-shop scheduling the two jobs are allowed to exchange machines in a feasible solution. In train

scheduling this refers to two trains traveling in opposite directions and exchanging adjacent track sections. Since this is physically not possible, a schedule including deadlocks is infeasible in train scheduling. Following the majority of the literature this type of infeasibility is disregarded here to simplify the optimization process.
Using the standard three-parameter classification, the single-track train scheduling problem is interpreted as a scheduling problem

$$Jm \mid r_i, d_i, recrc, block \mid \sum T_i$$

with the following conditions implied:

- A job can only be processed on one machine at a time.
- A machine can only process one job at a time.
- The processing of a job on a machine cannot be interrupted.
- The input data such as release times and processing times are assumed to be positive integers.

4 Overview of alternative modelings

While transforming and modeling the train scheduling problem as a job shop problem, two fundamental decisions have to be made. In the transformation phase the question arises how track sections with parallel tracks are to be considered. Three main approaches given in the literature are the *Parallel-Machine Approach* (see e.g. D'Ariano et al. (2007)), the *Machine-Unit Approach* (see e.g. Liu and Kozan (2009)) and the *Buffer Approach* (see e.g. Burdett and Kozan (2010)). The following Subsection 4.1 gives detailed explanations regarding these approaches, where only the first two approaches will be applied during the computational study in this paper. When dealing with the modeling phase, schedule defining decision variables are to be derived. Binary decision variables are introduced to model order dependencies and their determination is sufficient for a unique definition of the schedule. In Pan (1997) an overview of three well-known types of such binary variables is presented. This paper will focus on *precedence variables* and *order variables*, which are explained in more detail in Subsection 4.2.

4.1 Transformation of parallel tracks

The decision on how to transfer parallel tracks to machines implies the inclusion or exclusion of routing flexibility in stations and sidings. Figure 2 presents three main approaches to representing track sections with parallel tracks.

Following the Parallel-Machine Approach (PMA), each track is transferred to one machine M_k and by the definition of a train route one of the parallel tracks has to be chosen in advance. This means that the PMA excludes routing flexibility in stations and sidings. In contrast, by

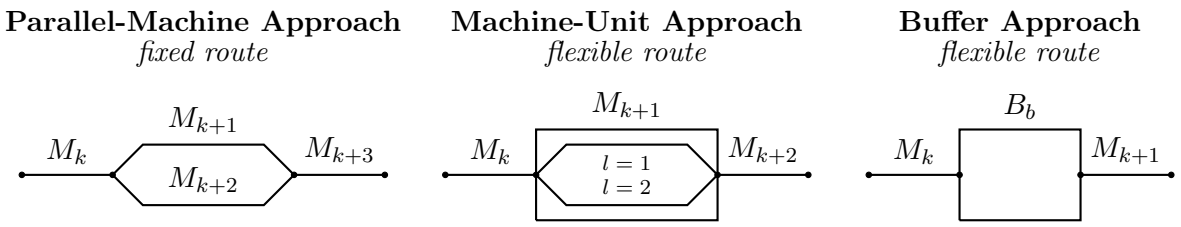

Figure 2: Representation of parallel tracks

applying the Machine-Unit Approach (MUA) parallel tracks are transformed to one machine M_k with certain machine units l, where the number of units equals the number of parallel tracks. Thus the route of each train defines the machines on which processing is to take place and the assignment of trains to machine units is done within the optimization process.

In a third alternative transformation, the Buffer Approach (BA), single tracks are transferred to machines and sections with parallel tracks are set to be buffers with a capacity equal to the number of parallel tracks. Train routes are defined by an ordered set of machines; stations and sidings are interpreted as black boxes without any regard to the structure inside. The platform and track assignment in stations and sidings has to be done afterwards according to the given solution of the scheduling problem. This means that routing flexibility is included in the MUA and the BA, where the actual routing decision is made in different planning phases.

Several aspects have to be considered when determining how to integrate routing decisions in stations and sidings in the optimization process. In the PMA these decisions are given with the train routes and taken as input data. This means that the platform assignment in stations and the track assignment in sidings have to be done without information on the starting times of the trains on adjacent track sections. Since an optimal schedule of all trains in the network is dependent on these routing decisions, the minimum total tardiness of all trains might be affected in advance.

Applying the BA the routing decisions do not have to be given as input to the problem, but have to be made based on fixed starting times of trains on tracks out of the scheduling process. Since an assignment with regard to the given starting times might be impossible, the BA implies no guarantee for the feasibility of an overall schedule.

Thus the advantage of the MUA as an integrated approach is an overall optimization with a guaranteed feasible schedule and minimum total tardiness. Nevertheless, the PMA might be reasonable in practice, since the exclusion of flexibility and additional decisions reduces complexity and size of the scheduling problem.

The resulting tradeoff between minimum total tardiness and computation time is discussed with regard to the application of different transformation approaches and their mathematical models in the following computational study. Since the input data and structure of models based on the BA differs from mathematical models representing PMA and MUA, the BA is excluded here and will be examined in future studies. It is expected that the objective function values reached by the integrated MUA are at least as good as those reached by the PMA, whereas the computation times are supposed to behave oppositely.

4.2 Application of different decision variables

The models developed to solve job-shop scheduling problems include assignment variables, which define order relations between operations of different jobs processed on the same machine. The two well-known types of assignment variables applied in this paper are *precedence variables* $y_{ii'k}$ and *order variables* x^r_{ik}, first implemented in Manne (1960) and Wagner (1959), respectively. Since recirculation is permitted, the variables have to be derived operation-based as follows:

$$y_{iji'j'k} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled before } O_{i'j'} \\ & \text{on machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

$$x^r_{ijk} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled as the } r\text{-th operation} \\ & \text{on machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

Both types of variables can be interpreted as orientating undirected edges in a disjunctive graph, which is a well-known representation of the job-shop scheduling problem. In Figure 3 an example with three operations of different jobs requiring the same machine is given. The arcs in the graph represent precedence relations between operations, where horizontal arcs define technological orders of the operations of the same job. All operations which have to be processed on the same machine are connected by undirected edges at first. A feasible solution is defined by an acyclic directed graph. The decisions made by setting one assignment variable equal to 0 or 1 are illustrated by the arcs in bold print, respectively.

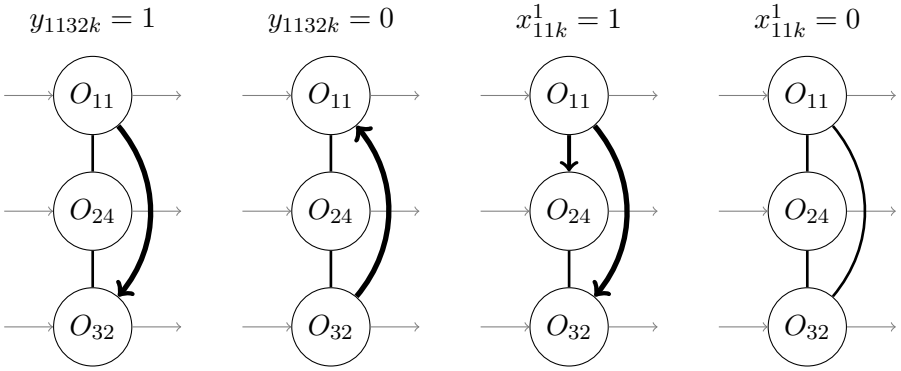

Figure 3: Application of assignment variables

The order of both operations O_{ij} and $O_{i'j'}$ on machine M_k is defined by determining one precedence variable $y_{iji'j'k}$ to be 0 or 1. This is shown by the two subgraphs on the left, in which the undirected edge connecting O_{11} and O_{32} is oriented defining the order of the operations depending on the value of variable y_{1132k}.
The subgraphs on the right illustrate best case and worst case decisions made by the determination of order variables x^r_{ijk}. Setting x^1_{11k} equal to 1 and with it operation O_{11} to be processed on M_k first determines all undirected edges connected to O_{11} to be arcs. In contrast, without

further information on other variables, the definition of x^1_{11k} to be equal to 0 determines no order relation between O_{11} and the other operations at all. This shows that the number of order relations set by one order variable depends on the order position considered and fixed variables on the same machine.
A third well-known type of assignment variables are binary time-indexed variables defined as follows:

$$z^t_{ik} := \begin{cases} 1 & \text{if job } J_i \text{ is scheduled on machine } M_k \text{ during time} \\ & \text{period } t \\ 0 & \text{otherwise.} \end{cases}$$

In contrast to precedence and order variables, time-indexed assignment variables are not directly related to the concept of a disjunctive graph. Including discrete time units implies a modeling structure significantly different from models based on the assignment variables presented above. Furthermore, time-based models are not easily comparable by means of computation time due to the dependency of the problem size on the input data such as processing times and release times. This is why the following computational study is focused on the comparison of precedence and order variables in terms of the number of variables and constraints needed to express the problem, computation time and problems solvable to optimality.

5 MIP formulations for job-shop scheduling problems with blocking based on the PMA

5.1 Notation and input data

In the following two subsections, the job-shop scheduling problem with blocking is modeled based on the PMA including fixed routes for all trains. There is a set of operations $\mathcal{O}^i$ given for every job J_i, and the required machine M_k and the processing time p_{ij} are known for every operation O_{ij}. Thus it is possible to determine sets of operations $OpMa^k$ for each machine M_k, which consist of all operations processed on this machine. Furthermore, the release time r_i and the due time d_i of every job J_i are given as input data to the problem. As mentioned above, precedence variables and order variables are defined and applied as assignment variables.
In order to derive the mathematical formulation using order variables, a set of order positions $\mathcal{R}^k$ is defined for every machine. Each of these sets consists of order positions $r = 1, \ldots, R_k$, where R_k specifies the number of order positions on machine M_k. It equals the number of operations to be processed on the machine to assure that order positions are consecutively assigned.

5.2 PMA-MIP formulation with precedence variables

The job-shop scheduling problem with blocking constraints and the minimization of total tardiness can be modeled based on the Parallel-Machine Approach, applying precedence variables as follows:

$$\sum_{i=1}^{n} T_i \qquad \longrightarrow \min! \tag{5.1}$$

s.t.

$$s_{ij} + p_{ij} \leq s_{ij+1} \qquad O_{ij} \in \mathcal{O}^i \setminus \{O_{in_i}\}, J_i \in \mathcal{J} \tag{5.2}$$

$$r_i \leq s_{i1} \qquad J_i \in \mathcal{J} \tag{5.3}$$

$$s_{in_i} + p_{in_i} = C_i \qquad J_i \in \mathcal{J} \tag{5.4}$$

$$T_i \geq C_i - d_i \qquad J_i \in \mathcal{J} \tag{5.5}$$

$$T_i \geq 0 \qquad J_i \in \mathcal{J} \tag{5.6}$$

$$y_{iji'j'k} + y_{i'j'ijk} = 1 \qquad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i < i', M_k \in \mathcal{M} \tag{5.7}$$

$$s_{i'j'} + M(1 - y_{iji'j'k}) \geq s_{ij} + p_{ij} \qquad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i', M_k \in \mathcal{M} \tag{5.8}$$

$$s_{i'j'} + M(1 - y_{iji'j'k}) \geq s_{ij+1} \qquad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i' \text{ and } j \neq n_i, M_k \in \mathcal{M} \tag{5.9}$$

$$y_{iji'j'k} \in \{0, 1\} \qquad O_{ij}, O_{i'j'} \in \mathcal{O}^k \text{ with } i \neq i', M_k \in \mathcal{M} \tag{5.10}$$

(5.1) describes the optimization criterion. In the set of constraints, inequality (5.2) implements the technological orders of all jobs by assuring each operation not be started earlier than the end of the processing time of the preceding operation of the same job. Inequality (5.3) guarantees the observance of the release times in restricting the starting time s_{ij} of the first operation of each job. The completion time C_i of each job is defined to be equal to the end of the processing time of its last operation O_{in_i} by equality (5.4). Constraints (5.5) and (5.6) determine the tardiness T_i of each job by the use of the completion time and the given due time as a true delay or zero. Equation (5.7) assures the uniqueness, completeness and exclusiveness of the precedence variables. For each pair of operations of different jobs requiring the same machine exactly one order relation, implying predecessor and successor, has to be defined. Following this order relation, the starting time of the succeeding operation has to be greater than or equal to the sum of the starting time and the processing time of the preceding operation as determined by inequality (5.8). Since a certain starting time relation holds only if the corresponding order relation of the operations is chosen, there are two constraints implemented for each pair of operations, where M is a sufficiently large positive constant. One of these constraints is always redundant in the determination of the solution.

The blocking constraint given by inequality (5.9) restricts the starting of a succeeding operation to be greater than or equal to the starting time of the successor of the preceding operation on the machine considered. Thus it is guaranteed that the preceding operation has left the machine before the succeeding operation starts.

Constraint (5.10) defines the assignment variables to be binary. The non-negativity of the remaining decision variables, in particular the operation starting times s_{ij}, the job completion times C_i and the job tardiness times T_i, is automatically given by constraints (5.2) to (5.6), since the input data are given as positive integers. In many studies dealing with mathematical programming formulations of scheduling problems, decision variables representing points in time

are additionally restricted to be positive integers, which leads to an increased computational effort. Since the modeling approaches will mainly be compared by means of computation time, the minimum number of integrality constraints is included to avoid running times being excessively influenced by them.

5.3 PMA-MIP formulation with order variables

Alternatively, the scheduling problem can be modeled applying order variables as follows:

$$\sum_{i=1}^{n} T_i \qquad \longrightarrow \min! \tag{5.11}$$

s.t.

constraints (5.2) to (5.6)

$$\sum_{r=1}^{R_k} x_{ijk}^r = 1 \qquad O_{ij} \in OpMa^k, M_k \in \mathcal{M} \tag{5.12}$$

$$\sum_{O_{ij} \in OpMa^k} x_{ijk}^r \leq 1 \qquad r \in \mathcal{R}^k, M_k \in \mathcal{M} \tag{5.13}$$

$$s_{i'j'} + M(1 - x_{ijk}^r) + M(1 - x_{i'j'k}^{r+1}) \geq s_{ij} + p_{ij} \quad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i' \text{ and } j \neq n_i, \; r \in \mathcal{R}^k \setminus \{R_k\}, M_k \in \mathcal{M} \tag{5.14}$$

$$s_{i'j'} + M(1 - x_{ijk}^r) + M(1 - x_{i'j'k}^{r+1}) \geq s_{ij+1} \quad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i' \text{ and } j \neq n_i, \; r \in \mathcal{R}^k \setminus \{R_k\}, M_k \in \mathcal{M} \tag{5.15}$$

$$x_{ijk}^r \in \{0, 1\} \qquad O_{ij} \in OpMa^k, r \in \mathcal{R}^k, M_k \in \mathcal{M} \tag{5.16}$$

The optimization criterion in (5.11) as well as the constraints regarding technological orders, release times, completion times and tardiness calculations are applied as explained in the previous section.

Equality (5.12) verifies that each operation having to be processed on a certain machine is assigned to exactly one order position on this machine. Similarly, inequality (5.13) assures that every order position is assigned to at most one operation. Considering the fact that the number of order positions of a machine equals the number of operations requiring this machine, the inequality will always be fulfilled with equality for a feasible solution.

Inequality (5.14) implements the order of the operations on the same machine by determining the starting time of the succeeding operation, which is not to be earlier than the end of processing of the preceding operation. This relation has to hold for each pair of operations assigned to consecutive order positions r and $r + 1$. Following the same structure, the blocking constraint is given by inequality (5.15). Letting O_{ij} and $O_{i'j'}$ be assigned to the positions r and $r + 1$, respectively, the processing of operation $O_{i'j'}$ can only start when job J_i has left M_k and operation O_{ij+1} is processed on another machine.

Constraint (5.16) defines the order variables to be binary. The non-negativity of all the other decision variables is implied as explained in 5.2.

6 MIP formulations for job-shop scheduling problems with blocking based on the MUA

6.1 Notation and input data

The Machine-Unit Approach basically requires the same input data as the Parallel-Machine Approach. The jobs are given with their technological orders, release and due times as well as their operations with processing times. The number of machine units has to be additionally given for each machine and some (re)definitions have to be made according to the sets and variables. The MUA transforms parallel tracks into one machine M_k with a corresponding set of machine units $\mathcal{L}^k = \{l \mid l = 1, \ldots, L_k\}$. The assignment of an operation requiring the machine to a certain machine unit is integrated into the optimization process. For this reason additional decision variables are defined as:

$$x_{ijkl} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is processed on unit } l \text{ on machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

It is implied that precedence and order variables have to be extended as follows:

$$y_{iji'j'kl} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled before } O_{i'j'} \text{ on unit } l \text{ on machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

$$x^r_{ijkl} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled as the } r\text{-th operation on unit } l \text{ on machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

The assignment variables x_{ijkl} are strongly related to the precedence and order variables respectively, since order relations are only to be defined for all pairs of operations being assigned to the same machine unit.
Furthermore, the number of order positions needed on a machine can no longer be determined by simply taking the number of operations requiring the machine. Since the assignment is done during the optimization process, each of the machine units has to have as many order positions available as there are operations requiring the machine. Therefore, the set of order positions for each machine unit is given as $\mathcal{R}^{kl} = \{r \mid r = 1, \ldots, R_{kl}\}$, where $R_{kl} = |OpMa^k|$.

6.2 MUA-MIP formulation with precedence variables

The job-shop scheduling problem with blocking constraints and the minimization of the total tardiness can be modeled based on the Machine-Unit Approach, applying precedence variables as follows:

$$\sum_{i=1}^{n} T_i \qquad \longrightarrow \min! \tag{6.1}$$

s.t.

constraints (5.2) to (5.6)

$$\sum_{l=1}^{m_k} x_{ijkl} = 1 \qquad O_{ij} \in OpMa^k, M_k \in \mathcal{M} \tag{6.2}$$

$$x_{ijkl} - y_{iji'j'kl} - y_{i'j'ijkl} \geq 0 \qquad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i < i', l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.3}$$

$$x_{ijkl} + x_{i'j'kl} - y_{iji'j'kl} - y_{i'j'ijkl} \leq 1 \qquad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i < i', l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.4}$$

$$s_{i'j'} + M(1 - y_{iji'j'kl}) \geq s_{ij} + p_{ij} \quad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i', l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.5}$$

$$s_{i'j'} + M(1 - y_{iji'j'kl}) \geq s_{ij+1} \qquad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i' \text{ and } j \neq n_i, \; l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.6}$$

$$x_{ijkl} \in \{0, 1\} \qquad O_{ij} \in OpMa^k, l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.7}$$

$$y_{iji'j'kl} \in \{0, 1\} \qquad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i', l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.8}$$

(6.1) defines the minimization of the total tardiness to be the optimization criterion. Constraints (5.2) to (5.6) are applied as given in Section 5.2.

Equation (6.2) ensures the unique and complete assignment of operations to machine units. Each of the operations requiring a certain machine is assigned to exactly one machine unit on this machine. Inequality (6.3) restricts the precedence variables related to a certain operation O_{ij} and a machine unit l to be equal to 0 if $x_{ijkl} = 0$ and operation O_{ij} is not assigned to machine unit l. Otherwise, exactly one precedence relation can be determined between O_{ij} and any other operation $O_{i'j'}$ on the same machine. It is sufficient to set up inequality (6.3) for pairs of operations with $i < i'$, since with this each pair of operations on the machine unit is considered exactly once and therefore all necessary precedence variables are allowed to be set to 1. Since constraint (6.3) only allows the relevant precedence variables to be set to 1, inequality (6.4) is set up to force one precedence relation to hold in case two operations are processed on the same machine unit.

Inequalities (6.5) and (6.6) refer to the order of processing and the blocking situation as explained in Section 5.2. The only difference is that these constraints are set up for each pair of operations being processed on the same machine unit l on M_k and not just on the same machine. Finally, the assignment variables x_{ijkl} and the precedence variables are defined to be binary in constraints (6.7) and (6.8).

6.3 MUA-MIP formulation with order variables

Alternatively, the scheduling problem can be modeled applying order variables as follows:

$$\sum_{i=1}^{n} T_i \qquad \longrightarrow \min! \tag{6.9}$$

s.t.

constraints (5.2) to (5.6)

$$\sum_{r=1}^{R_{kl}} \sum_{l=1}^{m_k} x_{ijkl}^{r} = 1 \qquad O_{ij} \in OpMa^k, M_k \in \mathcal{M} \tag{6.10}$$

$$\sum_{O_{ij} \in OpMa^k} x_{ijkl}^{r} \leq 1 \qquad r \in \mathcal{R}^{kl}, l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.11}$$

$$\left(\sum_{O_{ij} \in OpMa^k} x_{ijkl}^{r} \right) - \left(\sum_{O_{ij} \in OpMa^k} x_{ijkl}^{r+1} \right) \geq 0 \qquad r \in \mathcal{R}^{kl} \setminus \{R_{kl}\}, l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.12}$$

$$s_{i'j'} + M(1 - x_{ijkl}^{r}) + M(1 - x_{i'j'kl}^{r+1}) \geq s_{ij} + p_{ij} \quad O_{ij}, O_{i'j'} \in OpMa^k, r \in \mathcal{R}^{kl} \setminus \{R_{kl}\}, l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.13}$$

$$s_{i'j'} + M(1 - x_{ijkl}^{r}) + M(1 - x_{i'j'kl}^{r+1}) \geq s_{ij+1} \quad O_{ij}, O_{i'j'} \in OpMa^k \text{ with } i \neq i' \text{ and } j < n_i, r \in \mathcal{R}^{kl} \setminus \{R_{kl}\}, l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.14}$$

$$x_{ijkl}^{r} \in \{0, 1\} \qquad O_{ij} \in OpMa^k, r \in \mathcal{R}^{kl}, l \in \mathcal{L}^k, M_k \in \mathcal{M} \tag{6.15}$$

The optimization criterion as well as the constraints regarding technological orders, release, completion and tardiness times are applied as given in Section 5.2. In the following, the assignment of operations to order positions on machines is implemented again.

Constraint (6.10) assures that every operation is assigned to exactly one order position on exactly one machine unit of the required machine. As a complement inequality (6.11) guarantees that every order position is assigned to at most one operation. Inequality (6.12) defines the order position assignment to start at $r = 1$ and be consecutively without any free order position in front of an assigned position. These conditions have to be fulfilled to verify the correct implementation of the starting time relations and the blocking conditions, since they are valid for pairs of succeeding operations on the same machine unit. In the PMA-based formulation in Section 5.3 inequality (6.12) is automatically imposed by setting $R_k = |OpMa^k|$.

As already mentioned, the starting time relation set by constraint (6.13) and the blocking constraint given by inequality (6.14) must be satisfied for each pair of operations assigned to the same machine unit. Constraint (6.15) defines the order variables to be binary.

7 Computational study

7.1 Test instances

The test instances with different numbers of jobs and machines are generated randomly from an underlying data structure. The single-track network introduced in Section 2 is tested with different numbers of trains traveling. Applying the PMA and the MUA, this network can be transformed into a set of eleven and nine machines respectively as shown in Figure 4.

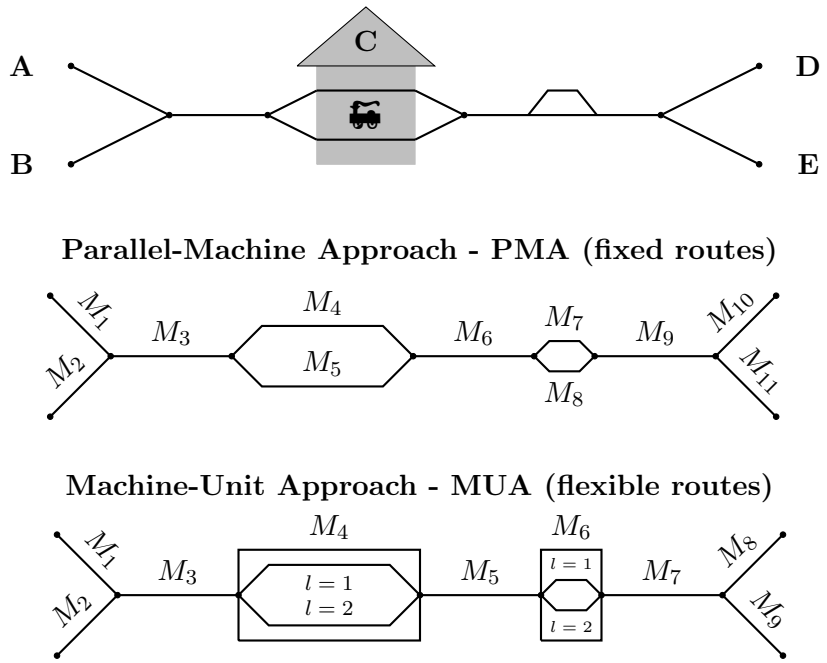

Figure 4: Single-track network transformation

Trains can only enter and leave the network at the bounding nodes A, B, D and E. The train routes are generated randomly including shortest path routes and routes in which station C is used as a terminal station. Three of these routes are exemplary given in Figure 5. The train routes are interpreted as technological orders of the jobs J_i and their release dates r_i and due dates d_i generated randomly as follows:

$$r_i \in \left[0\ ,\ 1.5 \min_{J_i \in \mathcal{J}} \left\{ \sum_{j=1}^{n_i} p_{ij} \right\}\right] \quad \text{and} \quad d_i = \left\lceil r_i + \left(1.2 \sum_{j=1}^{n_i} p_{ij} \right) \right\rceil$$

The determination of the processing times p_{ij} for the operations O_{ij} of the jobs J_i accounts for the underlying idea that trains do not have random travel times. Three characteristic train types are introduced, namely passenger, express and freight trains, with corresponding travel times on single tracks, in stations and sidings. Main differences such as speed and expected passenger transfer time in stations are thereby considered. The processing times p_{ij} are assigned to the operations according to the train type chosen for the job, where this choice is randomly

done. The only restriction applied is that each train type has to be chosen at least once in every instance.

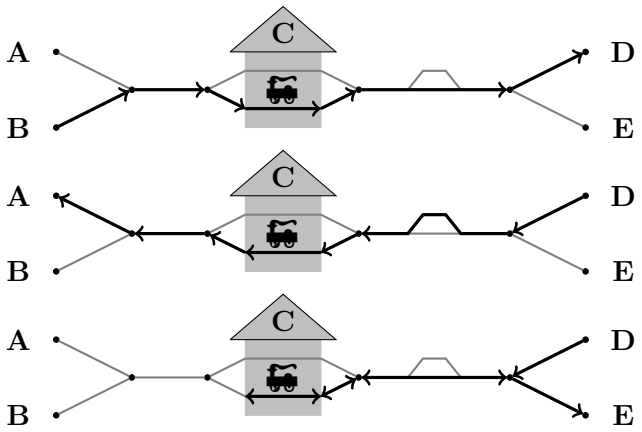

Figure 5: Three possible train routes

7.2 Computational results

The instances generated consist of 10, 15 and 20 trains traveling through the given single-track network. In other words, the test instances include 10, 15 and 20 jobs to be processed on 11 machines in the Parallel-Machine Approach and 9 machines in the Machine-Units Approach. To avoid the effect of particular characteristics in the comparison, five instances are randomly generated and tested for each number of jobs. Additionally, a small instance with 6 jobs was constructed to run the first tests on the models.

Tables 1 and 2 summarize the computational results obtained by the mixed integer programming formulations of the PMA and the MUA, respectively. The first column consists of the instance size (n, m), where n denotes the number of jobs and m denotes the number of machines. Column two gives an instance index. Columns three to six contain the number of variables (#var) and the number of constraints (#con) of the corresponding MIP formulation, the calculated total tardiness ($\sum T_i$) and the computation time needed to solve mathematical models based on the precedence variables $y_{iji'j'k}$. In the same pattern columns seven to ten include the information on computations done with mathematical models based on the order variables x^r_{ijk}. Total tardiness values denoted with asterisk are proven to be optimal. For all values without proven optimality the percentage gap between the best known integer solution and the lower bound is additionally given.

The instance 4 of size $(10, 11)$ was run without a time limit. After 24 hours of computation there was still no integer solution found for the order-variable-based formulation. Due to this, a limit on computation time was set to two hours, since the main aspects of the solving procedure characteristics can also be examined based on these results.

Instance		**Precedence variables**				**Order variables**			
(n, m)	no.	**#var**	**#con**	$\sum T_i$	Time	**#var**	**#con**	$\sum T_i$	Time
$(6, 11)$	1	186	372	17*	0.56s	236	1338	17*	6.75s
$(10, 11)$	1	530	1146	138*	5.8s	608	7014	-	2 h
$(10, 11)$	2	498	1072	90*	1.39s	568	6283	-	2 h
$(10, 11)$	3	518	1120	72*	0.97s	596	6730	-	2 h
$(10, 11)$	4	558	1215	41*	0.62s	644	7868	-	2 h
$(10, 11)$	5	490	1052	71*	1.33s	568	6235	-	2 h
Mean:		**518.8**	**1121**		**2.02s**	**596.8**	**6826**		
$(15, 11)$	1	1145	2564	88*	643.83 s (10.7 min)	1266	23505	-	2 h
$(15, 11)$	2	1199	2689	172*	3368.22 (56.1 min)	1328	25652	-	2 h
$(15, 11)$	3	1149	2565	163 (23.63%)	2 h	1270	23552	-	2 h
$(15, 11)$	4	1189	2668	161 (32.55%)	2 h	1318	25292	-	2 h
$(15, 11)$	5	1115	2488	97*	1252 s (20.86 min)	1288	22455	-	2 h
Mean:		**1159.4**	**2594.8**			**1294**	**24091.2**		
$(20, 11)$	1	1994	4532	372 (75.75%)	2 h	2142	55316	-	2 h
$(20, 11)$	2	2038	4643	393 (75.81%)	2 h	2168	57140	-	2 h
$(20, 11)$	3	2106	4802	426 (77.19%)	2 h	2262	60674	-	2 h
$(20, 11)$	4	2036	4628	509 (83.30%)	2 h	2200	57400	-	2 h
$(20, 11)$	5	2084	4755	375 (76.91%)	2 h	2256	60252	-	2 h
Mean:		**2051.6**	**4672**			**2205.6**	**58156.4**		

Table 1: Mixed integer programming results for the Parallel-Machine Approach

From Table 1 and Table 2 the following five comparisons can be made:

- comparison of precedence and order variables in terms of the size of the formulation,
- comparison of precedence and order variables in terms of computation time,
- comparison of PMA and MUA in terms of objective function values,
- comparison of PMA and MUA in terms of the size of the formulation,
- comparison of PMA and MUA in terms of computation time.

For each instance category with $n \in \{10, 15, 20\}$ the mean number of variables as well as the mean number of constraints are given in separate rows written in bold face. *Comparing precedence and*

Instance		**Precedence variables**				**Order variables**			
(n, m)	no.	**#var**	**#con**	$\sum T_i$	Time	**#var**	**#con**	$\sum T_i$	Time
$(6, 9)$	1	440	756	8*	1.22s	342	2549	8*	20.53s
$(10, 9)$	1	1240	2240	80*	2187.49 s (36.45 min)	898	12937	-	2 h
$(10, 9)$	2	1167	2089	83*	1127.9 s (18.8 min)	835	11526	-	2 h
$(10, 9)$	3	1232	2232	61*	153.55 s (2.56 min)	894	12867	-	2 h
$(10, 9)$	4	1313	2402	27*	7.97 s	963	14749	-	2 h
$(10, 9)$	5	1159	2071	42*	24.07 s	839	11550	-	2 h
Mean:		**1222.2**	**2206.8**		**700.2 s** (11.67 min)	**885.8**	**12725.8**		
$(15, 9)$	1	2738	5094	65 (68.16%)	2 h	1940	45161	-	2 h
$(15, 9)$	2	2867	5361	161 (71.42%)	2 h	2043	49410	-	2 h
$(15, 9)$	3	2738	5085	145 (70.15%)	2 h	1940	45040	-	2 h
$(15, 9)$	4	2855	5347	132 (80.16%)	2 h	2037	49218	-	2 h
$(15, 9)$	5	2637	4864	71 (42.71%)	2 h	1853	41851	-	2 h
Mean:		**2767**	**5150.2**			**1962.6**	**46136**		
$(20, 9)$	1	4735	8860	356 (88.16%)	2 h	3277	103733	-	2 h
$(20, 9)$	2	4868	9174	411 (92.85%)	2 h	3382	109505	-	2 h
$(20, 9)$	3	5049	9548	514 (93.19%)	2 h	3521	117289	-	2 h
$(20, 9)$	4	4868	9152	490 (94.37%)	2 h	3398	109761	-	2 h
$(20, 9)$	5	5001	9472	459 (96.80%)	2 h	3513	116615	-	2 h
Mean:		**4904.2**	**9241.2**			**3418.2**	**111380.6**		

Table 2: Mixed integer programming results for the Machine-Unit Approach

order variables by means of the size of the formulations, it can be seen that order-variable-based formulations include more constraints than precedence-variable-based models independent of the approach used. The multiplier between the numbers of constraints in the formulations increases with the number of jobs considered. While in the PMA (Table 1) the order-variable-based model with 10 jobs has approximately 6 times as many constraints as the precedence-variable-based one, this increases to about 9 times as many for the instances with 15 jobs, and to about 12 times as many for the 20-job-instances. The same pattern can be observed for the number of

constraints in the MUA (Table 2).
Regarding the number of variables, it can be observed that when using the PMA the order-variable-based formulation consists of more variables than the precedence-variable-based one, whereas the opposite is true for the models applying the MUA. The number of variables increases with the number of jobs included in the instance, where the amount of variables grows over-proportionally.
In line with the size of the formulations the computation time of solving models with precedence variables is significantly lesser than the time needed to solve problems modeled with order variables. More particularly, there are no integer solutions found within a time frame of two hours for order-variable-based formulations, when the number of jobs is greater than or equal to 10. In terms of computation time the precedence variables clearly outperform the order variables. This seems to be due to structural reasons, since even for the order-variable-based formulations using the MUA, which include less variables than their counterparts, there are no integer solutions found.
In the literature there are different strategies to help solvers deal with order variables and improve run time results. However, since the difference in computation time is considerable for all of the test instances, results comparable to those obtained with precedence variables are not to be expected.

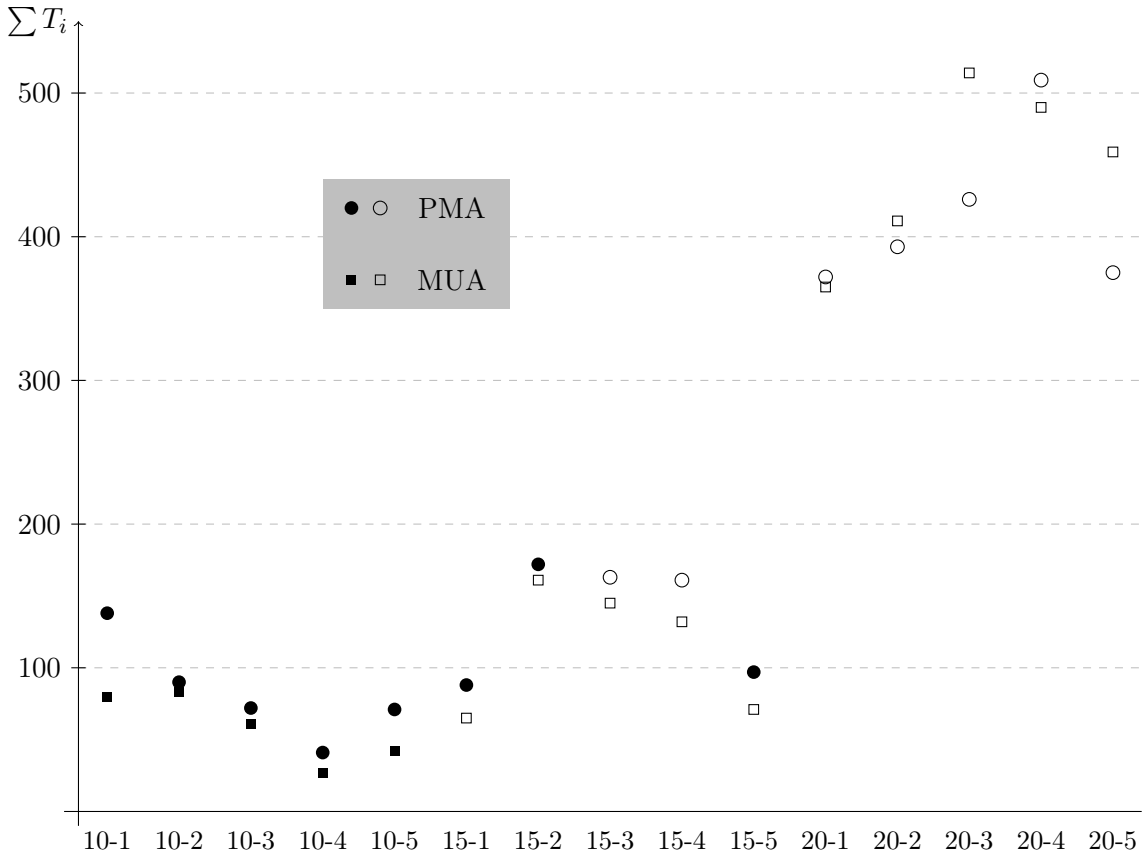

Figure 6: Comparison of PMA and MUA in terms of total tardiness

A comparison of the Parallel-Machine Approach and the Machine-Unit Approach can be done with respect to objective function values, formulation size and computation time. Figure 6 illustrates the objective function values achieved by applying the PMA and the MUA with precedence variables, which are given in detail in column five of Tables 1 and 2 respectively. Results denoted with filled symbols are proven to be optimal, whereas drawn symbols represent the best known integer solution after two hours of computation.

As expected, the additional routing flexibility and with it the optimal machine assignment in sections with parallel machines lead to smaller total tardiness values in the MUA (squares). It is remarkable, that for most of the larger instances the results without proven optimality of the MUA show smaller objective function values than those achieved by the random machine selection in the PMA (circles). This aspect might be used in future research to balance the tradeoff between optimality and computation time. An idea might be to design a procedure, which determines near optimal machine assignments for sections with parallel tracks by solving a MIP formulation and constructs the remaining parts of the solution heuristically.

Considering the formulation size, it can be stated that the models applying the Machine-Units Approach are about twice as large as models applying the Parallel-Machine Approach. Comparing the mean number of variables and the mean number of constraints of precedence-variable-based models, the number of variables multiplies by a factor of about 2.4 from the PMA to the MUA formulation, while the number of constraints increases by a factor of about 2.0. For order-variable-based models the mean number of variables grows by a multiplier of approximately 1.5 and the mean number of constraints multiplies by a factor of about 1.9. All in all, there is a significant difference in the formulation size between PMA and MUA, where the differences in the mean numbers of variables and constraints are larger for models based on precedence variables.

As already indicated by the size of the formulations, the computation time to solve models applying the MUA is greater than the time needed to solve models applying the PMA. The following comparisons are made only by regarding the results of precedence-variable-based formulations, since there is no integer solution found for the order-variable-based models. The only reasonable comparison of mean run times can be done for the test instances with 10 jobs. It is given in column six of Tables 1 and 2 that 10-job instances are solved to optimality in 2.02 s on average with a random, but fixed machine assignment and in 11.67 min on average with an integrated flexible machine assignment.

For larger instances comparisons are drawn for the number of instances solved to optimality in less than two hours and for the remaining gap between best known integer solution and lower bound. Applying the PMA, 3 of the 15-job instances are solved to optimality and the average gap for 20-job instances is about 77.79%. On the contrary, applying the MUA none of the 15-job instances is solved to optimality within two hours and the average gap for 20-job instances is about 93.07%. These results emphasize the tradeoff between routing flexibility and with it the lowest possible total tardiness values and computation time.

One last observation refers to the results obtained when solving different, randomly generated instances with the same number of jobs and machines. The most extreme differences are found when applying the PMA with precedence variables to the instances with 15 jobs and in calcu-

lating solutions using the MUA with precedence variables for problems with 10 jobs. For the instances of size $(15, 11)$ the least computation time is realized by solving problem 1 in 10.7 min, while instances 3 and 4 could not be solved to optimality within two hours. Similarly, instance 4 of size $(10, 9)$ is solved to optimality within 7.97 s, while solving instance 1 of the same size lasts 36.45 min. It is supposed, that there is a structural reason for such differences in the computation times of models containing almost the same number of variables and constraints.

8 Extensions

As mentioned briefly in the introduction, there are many extensions, which can be applied to the models described above. From the train scheduling point of view it is crucial that trains of different types are transformed to equally important jobs. In minimizing the sum of tardiness times, it might come to pass that some trains are restricted to wait, so that the majority of trains can easily pass through and reach their desired arrival times. This means that a constant objective function value can be obtained by a balanced as well as an unbalanced appearance of tardiness of trains. The assignment of waiting times and with it potential tardiness times can be controlled implicitly by introducing job weights $w_i \in [0, 1]$ to declare priorities of trains, so that trains with a high priority are not supposed to have long travel times and large tardiness values. This refers to a change of the objective function to the weighted total tardiness

$$\sum_{J_i \in \mathcal{J}} w_i T_i \quad \rightarrow \min! \tag{8.1}$$

In line with this idea, there might also be high priority trains, which are supposed to reach their final destination as fast as possible and never be made to wait for other trains to pass. For these trains a no-wait condition of the following form can be included in the models.

$$s_{ij} + p_{ij} = s_{ij+1} \qquad \text{for all } O_{ij} \in \mathcal{O}^i, J_i \in \mathcal{J}^{no-wait} \tag{8.2}$$

Such a job-shop problem, including blocking and no-wait conditions with a makespan objective, has been examined and solved using a Branch and Bound procedure in Mascis and Pacciarelli (2002).

Another important aspect in train scheduling, which has been addressed in Oliveira and Smith (2000), is the meeting of trains for passengers to change. Integrating desired meetings of particular trains in a certain station with a minimum duration requires the introduction of a set of train pairs, which are supposed to meet, as well as parameters and constraints to assure the assignment of both trains to the right tracks at the right time.

In optimizing real-world train scheduling problems headways have to be included for reasons of safety as done in Burdett and Kozan (2010). To assure these minimum time frames between two succeeding trains on the same track, the direction of the trains, the train length and the end of their traversal have to be included in the models. Additionally, the tracks have to be transformed to machines, so that a machine represents the track section between two signals for which the headway is defined. Applying this transformation, the number of machines in the models will

increase and the formulations will end up being much more complex.
During the last years the ideas of ecological green transportation has increasingly gained interest. In order to integrate objectives such as minimal fuel consumption or lesser exhaust emissions, the velocity of the trains has to be taken into account within the formulations. This means that the speed of the trains becomes an additional decision variable and the travel time of a train in a track section is not constant and known in advance. A formulation including variable velocity is examined and heuristically solved in Liu and Kozan (2009).

9 Conclusion

In this computational study a comparison is drawn between the applications of a Parallel-Machine Approach and a Machine-Unit Approach using two types of decision variables. The Parallel-Machine Approach is based on a random assignment of a machine from a set of parallel machines, whereas the Machine-Unit Approach integrates the optimal choice of a machine in the optimization process.
As expected, models with an integrated routing flexibility generate better solutions in terms of total tardiness values, however clearly require more computation time to reach optimal solutions. There is a direct tradeoff between desirable objective function values and run time.
Both approaches were tested solving randomly generated instances with 11 (9) machines and up to 20 jobs. The numbers of variables and constraints increased over-proportionally with the number of jobs in the instance independent of the approach and the type of decision variables used. This seems to be the main reason for upcoming difficulties in finding optimal or even feasible solutions for larger test instances. Generally, it must be said that the test instances including 11 machines and 20 jobs are rather small in relation to real-world problems. This justifies the need to improve the MIP solving procedure and implement heuristic methods, which might be able to achieve near-optimal solutions in shorter run times.
Regarding the type of decision variables, precedence variables of the form $y_{iji'j'k}$ clearly outperform order variables x^r_{ijk} in the computational study independent of the approach used and the size of the instance. Since the number of constraints in order-variable-based models is considerably larger than in precedence-variable-based formulations and the amount of information carried by order-variable-settings can be very low as explained in Section 4.2, the construction of a feasible solution is extremely difficult. This seems to be why there is no feasible solution found for order-based-formulations with ten or more trains.
In future research, heuristic methods are to be studied to improve the computation time of the models with an emphasis on the Machine-Unit Approach. It is remarkable that for several instances the solution of the MUA without proven optimality shows a smaller total tardiness value than the optimal solution of the PMA. This aspect might be used as a basis to combine the MIP solving with a heuristic procedure to obtain near-optimal solutions.
Furthermore, the differences in computation time needed to solve instances of the same size indicate structural characteristics, which seem to make some instances harder to solve than others.

Attempts have been made in job-shop scheduling to characterize instances, where it remains to be seen, whether these conditions hold for job-shop problems with blocking as well. Such characteristics, once known, might also be used to help solvers in dealing with these types of problems.

A Appendix: Notation

Sets and indices

$\mathcal{J}$	$:= \{J_i \mid i = 1, 2, \ldots, n\}$	set of jobs (trains)
$\mathcal{L}^k$	$:= \{l \mid l = 1, 2, \ldots, m_k\}$	set of parallel machine units (track lines) of machine M_k
$\mathcal{M}$	$:= \{M_k \mid k = 1, 2, \ldots, m\}$	set of machines (track sections)
$\mathcal{M}^i$	$:= \{M_k \mid \exists\ rout_{ij} = k\}$	set of machines, on which job J_i has to be processed
$\mathcal{O}^i$	$:= \{O_{ij} \mid j = 1, 2, \ldots, n_i\}$	ordered set of operations of job J_i
$OpMa^k$	$:= \{O_{ij} \mid rout_{ij} = k\}$	set of operations having to be processed on machine M_k
$\mathcal{R}^k$	$:= \{r \mid r = 1, 2, \ldots, R_k\}$	set of order positions on machine M_k
$\mathcal{R}^{kl}$	$:= \{r \mid r = 1, 2, \ldots, R_{kl}\}$	set of order positions on unit l of machine M_k

Parameters

d_i	:= due time of job J_i, (desired arrival time of a train at its final destination)
p_{ij}	:= processing time of operation O_{ij}, (running time of a train in a track section)
r_i	:= release time of job J_i, (earliest departure time of a train at its starting station)
$rout_{ij}$	:= index of the machine M_k, on which operation O_{ij} has to be processed

Decision variables

C_i	:= completion time of job J_i, (arrival time of a train at its final destination)
T_i	:= tardiness of job J_i, (tardiness of a train at its final destination)
s_{ij}	:= starting time of operation O_{ij} of job J_i

$$x_{ijkl} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled on unit } l \text{ of machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

$$x^r_{ijk} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled as the } r\text{-th operation on machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

$$x^r_{ijkl} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled as the } r\text{-th operation on unit } l \text{ of machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

$$x^t_{ik} := \begin{cases} 1 & \text{if job } J_i \text{ is processed on machine } M_k \text{ in period } t \\ 0 & \text{otherwise.} \end{cases}$$

$$y_{iji'j'k} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled before } O_{i'j'} \text{ on machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

$$y_{iji'j'kl} := \begin{cases} 1 & \text{if operation } O_{ij} \text{ is scheduled before } O_{i'j'} \text{ on the } l\text{th unit of machine } M_k \\ 0 & \text{otherwise.} \end{cases}$$

References

Burdett, R. L. and E. Kozan, 2010: A disjunctive graph model and framework for constructing new train schedules. *European Journal of Operational Research*, **200 (1)**, 85–98.

Cacchiani, V., L. Galli, and P. Toth, 2013: A tutorial on train timetabling and train platforming problems. Tech. Rep. TR-13-10, Università di Pisa, Dipartimanto di Informatica.

Cai, X. and C. J. Goh, 1994: A fast heuristic for the train scheduling problem. *Computers & Operations Research*, **21 (5)**, 499 – 510.

Carey, M. and D. Lockwood, 1995: A model, algorithms and strategy for train pathing. *Journal of the Operational Research Society*, 988–1005.

Cordeau, J.-F., P. Toth, and D. Vigo, 1998: A survey of optimization models on train routing and scheduling. *Transportation science*, **32 (4)**, 380–404.

D'Ariano, A., D. Pacciarelli, and M. Pranzo, 2007: A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research*, **183 (2)**, 643–657.

Gholami, O., Y. N. Sotskov, and F. Werner, 2013: Fast edge-orientation heuristics for job-shop scheduling problems with applications to train scheduling. *International Journal of Operational Research*, **2**, 19–32.

Gröflin, H. and A. Klinkert, 2009: A new neighborhood and tabu search for the blocking job shop. *Discrete Applied Mathematics*, **157 (17)**, 3643–3655.

Higgins, A., E. Kozan, and L. Ferreira, 1997: Heuristic techniques for single line train scheduling. *Journal of Heuristics*, **3 (1)**, 43–62.

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

Kreuger, P., M. Carlsson, J. Olsson, T. Sjöland, and E. Åström, 1997: Trip scheduling on single track networks - the tuff train scheduler. *Workshop on Industrial Constraint Directed Scheduling*, 1–12.

Lamorgese, L. and C. Mannino, 2013: The track formulation for the train dispatching problem. *Electronic Notes in Discrete Mathematics*, **41**, 559 – 566.

Lange, J., 2015: Approaches to modeling job-shop problems with blocking constraints. *Proceedings of the 7th Multidisciplinary International Conference on Scheduling: Theory and Applications*, Prague, 645–648.

Liu, S. Q. and E. Kozan, 2009: Scheduling trains as a blocking parallel-machine job shop scheduling problem. *Computers & Operations Research*, **36 (10)**, 2840–2852.

Lusby, R. M., J. Larsen, M. Ehrgott, and D. Ryan, 2011: Railway track allocation: models and methods. *OR spectrum*, **33 (4)**, 843–883.

Manne, A. S., 1960: On the job-shop scheduling problem. *Operations Research*, **8 (2)**, 219–223.

Mascis, A. and D. Pacciarelli, 2002: Job-shop scheduling with blocking and no-wait constraints. *European Journal of Operational Research*, **143 (3)**, 498–517.

Oliveira, E. and B. M. Smith, 2000: A job-shop scheduling model for the single-track railway scheduling problem. Research Report Series 21, School of Computing, University of Leeds.

Pan, C.-H., 1997: A study of integer programming formulations for scheduling problems. *International journal of systems science*, **28 (1)**, 33–41.

Sauder, R. L. and W. M. Westerman, 1983: Computer aided train dispatching: decision support through optimization. *Interfaces*, **13 (6)**, 24–37.

Szpigel, B., 1973: Optimal train scheduling on a single track railway. *Operations Research '72*, M. Ross, Ed., North-Holland Publishing Company, International Federation of Operational Research Societies, 343–352.

Van Laarhoven, P. J., E. H. Aarts, and J. K. Lenstra, 1992: Job shop scheduling by simulated annealing. *Operations research*, **40 (1)**, 113–125.

Wagner, H. M., 1959: An integer linear-programming model for machine scheduling. *Naval Research Logistics Quarterly*, **6 (2)**, 131–140.