A multi-criteria master production scheduling approach for special purpose machinery

Christian Gahm*, Bastian Dünnwald, Ramin Sahamie

Augsburg University, Universitaetsstrasse 16, 86159 Augsburg, Germany

Keywords:
Master production schedule
Series production
Multi-criteria
Lead-time minimization
Resource leveling
Variable neighborhood search

A B S T R A C T

This paper presents a multi-criteria master production scheduling approach as the final assembly of special purpose machines is known to be very cost intensive. These costs are mainly influenced by the master production schedule (MPS). Two major cost drivers arise. First, long assembly lead-times (up to several months) combined with high product values result in high capital commitments; thus, lead-times need to be minimized. Moreover, the factory calendar must be considered while calculating the MPS because the factory calendar can significantly influence the resulting lead-times. Second, contractual penalties and compensation costs arise if confirmed delivery dates cannot be kept. Therefore, resource requirements must be accounted for, and an MPS that is executable on the assembly shop floor must be calculated. To increase planning flexibility, we do not restrict the resource utilization with a formal constraint; instead, we introduce the additional objective of resource leveling. Consequently, the conflicting objectives lead-time minimization and resource leveling are integrated into a single objective function, in which the decision maker's preferences are represented by a weighting factor. To calculate such an MPS, we develop a tailor-made construction heuristic combined with a randomized variable neighborhood descent procedure. We evaluate our solution method by solving small instances with a commercial solver and large-scale instances from an application case of an aerospace company. Our results reveal that the decision maker's preferences are adequately reflected by the weighting factor. Moreover, we can provide a rule of thumb for selecting an appropriate initial weighting factor.

1. Introduction

The master production scheduling problem addressed in this paper is based on an application case that originated in the aerospace industry. As many German machine and plant manufacturers have done, the company has reacted to the challenges resulting from globalization and changed market relations by individualizing and segmenting its products and services. The company now offers complex, customer-specific top-level products (also called special purpose machines) for certain markets. To provide customer-specific products within competitive delivery times, special purpose machines are manufactured using an assemble-to-order (ATO) strategy, whereas the assembly itself is organized as a series production. This ATO strategy and several characteristics of the product and assembly process make the final assembly the main focus of interest in achieving two fundamental goals: cost reduction and customer satisfaction. The effect of the final assembly on these two goals can be explained in two ways. First, long assembly lead-times (up to several months) combined with a high product value (up to several million Euros) lead to high capital commitments. The long assembly lead-times result from a high level of manual assembly effort, which is driven by the use of cutting-edge technology and the high complexity of the product. Second, the final assembly is directly linked to customer delivery and thus has a direct impact on customer satisfaction.

According to Vieira and Favaretto (2006), master production scheduling (also called master planning (MP); Rhode and Wagner, 2008) "[…] is a key decision-making activity, in which strategic goals from business planning are translated into an anticipated statement of production, from which all other schedules at lower levels are derived". The importance of MP becomes obvious when the interdependencies with other planning tasks in a hierarchical production planning system (HPPS) are analyzed, as affirmed by several authors (e.g., Vollmann et al., 2005; Rhode and Wagner, 2008). According to these authors, MP is the basic input for dependent planning tasks, such as capacity planning, production planning and scheduling, distribution and transport planning, sales planning, order promising or purchasing, and material requirements planning. The importance of MP is also noted in several publications on customer-oriented individual production and ATO, such as Drexl et al. (1994), Franck et al. (1997), and Hans et al. (2007).

The general task of MP that is pertinent to this paper is the determination of a master production schedule (MPS). An MPS is a

* Corresponding author. Tel.: +49 821 5984359; fax: +49 821 5984353.
E-mail addresses: christian.gahm@wiwi.uni-augsburg.de (C. Gahm), bastian.duennwald@wiwi.uni-augsburg.de (B. Dünnwald). ramin.sahamie@wiwi.uni-augsburg.de (R. Sahamie).

temporal framework that coordinates dependent planning tasks and the flow of materials. On the basis of this function and the characteristic planning environment of special purpose machinery, we define the determination of assembly start and completion dates for assembly orders as the basic task of MP. In contrast to other approaches concerning MP, we do not integrate additional planning tasks, such as material requirements planning, into our analysis because the required information is not available at the time of planning.

As MP has such an important coordination function and "the objective function drives the logic behind the execution" (Vieira and Favaretto, 2006), both fundamental goals must be represented by the planning objectives. In terms of cost reduction, an optimized MPS can influence capital commitments. Cost reduction is primarily achieved through lead-time minimization. Concerning the planning problem at hand, the lead-time can only be improved if factory calendars (also called break calendars; cf. Trautmann, 2001) significantly influence the planning result. This influence is important for the underlying MPS problem and also other scheduling problems, such as "[…] real-life projects, make-to-order production, or process flows in the chemical industries, […]" (Neumann et al., 2003). Generally, scheduling with break calendars is termed calendarization (introduced by Zhan (1992)) and is addressed, for example, by Schwindt and Trautmann (2000) and Franck et al. (2001). Calendarization contributes to customer satisfaction because shorter assembly lead-times reduce customer delivery times. Other significant cost factors are contractual penalties and compensation costs, which are directly linked to the second goal of customer satisfaction, particularly the objective of high delivery reliability. As customer satisfaction cannot be directly influenced by MP (confirmed delivery dates do not exist at the time of planning), resource leveling is defined as a surrogate objective. The purpose of resource leveling is to support the operability of the MPS on the assembly shop floor and thus to enable on-time delivery. Moreover, resource leveling allows workforce adjustment costs to be reduced or even completely avoided. To address these conflicting objectives, namely, lead-time minimization and resource leveling (the conflict will be discussed in the following sections), we developed a multi-criteria master production scheduling approach.

As the following section will show, the existing literature does not sufficiently address the problem at hand; therefore, a new planning approach is required.

2. Literature review

Capital commitments are one of the main cost drivers in the production of special purpose machinery; thus, their reduction is the primary objective of MP. However, these costs are directly linked to work-in-process inventory costs, and they are very difficult to assess because of the vast number of materials and their time of assembly. Therefore, an operational surrogate objective is used: lead-time minimization (or throughput time minimization – cf. Pinedo, 2009). In the literature, many different criteria are used to evaluate lead-time performance. Examples include the sum of (weighted) lead-times, mean (weighted) lead-time, maximal lead-time, sum of deviations, and mean deviations. Note that the first two criteria listed are pairwise equivalent. In addition, the maximal lead-time criterion is not suitable because only orders with long net lead-times would be affected by the optimization. Moreover, to our knowledge, no comprehensive study comparing these objectives exists; thus, no objective is claimed to be superior to the others. As a consequence, the suitable criterion depends on the specific problem and the decision maker's preferences. In this paper, the objective of lead-time minimization is represented by a lead-time deviation criterion that is equivalent to minimizing the mean lead-time.

The objective of resource leveling (also called resource balancing or resource smoothing) has gained increasingly more attention in recent years (cf. Anagnostopoulos and Koulinas, 2010; Drótos and Kis, 2011; Gather et al., 2011). Comprehensive introductions to this topic can be found in Younis and Saad (1996), Neumann and Zimmermann (1999), Caramia and Dell'Olmo (2006), and Ballestín et al. (2007). Anagnostopoulos and Koulinas (2010) state that the "[…] scheduling objective of resource leveling is to make the resource requirements as even as possible over the entire project horizon, usually, without explicit resource considerations to be taken into account". This statement is consistent with Drótos and Kis's (2011) statement that "in resource leveling problems the objective is to minimize a function of the resource utilization over time". Typically total squared utilization costs are minimized to achieve balanced resource utilization (Gather et al., 2011). Neumann and Zimmermann (1999) analyze the suitability of three different objective function classes for projects with minimum and maximum time lags. Following these contributions and the requirements of the underlying planning problem, we use a criterion defined as the root ("normalized") of the sum of squared deviations from a desired value to balance resource utilization. This function is used because it explicitly penalizes strong deviations.

Concerning conflicting objectives, a vast number of methods to solve conflicts exist in the literature (introductions, overviews, and details of these methods can be found e.g., in Gupta et al. (1991), Dyer et al. (1992), Keeney et al. (1993), Kirkwood (1997), Belton and Stewart (2002), Hoogeveen (2005), Figueira et al. (2005), or T'kindt and Billaut (2006)). Here, the challenge is to evaluate the different methods of multi-criteria decision analysis, multi-criteria decision making, multi-objective decision making (MODM), or multi-attribute decision making (MADM) with regard to their applicability and suitability for the given decision problem. As the number of general topics about decisions with multiple objectives suggests, this challenge may be substantial. A tentative guideline for method selection is given in Guitouni and Martel (1998). A first distinction can be made by the determination of alternatives (problem solutions). As an explicit determination of (discrete) alternatives is not applicable for the problem at hand, only methods with an implicit determination are of interest (this type of method is often categorized as an MODM method and is also called multiobjective programming (MOP); cf. Ehrgott and Gandibleux, 2000). One can also distinguish between methods where the (final) decision is made a posteriori or a priori. The a posteriori methods are based on a set of compromise solutions and their attributes (MADM; sometimes also called generate-first–choose-later or construction and exploitation methods). Some a posteriori methods are outranking-based methods or methods based on multi-attribute utility/value theory (cf. Dyer, 2005). The a priori methods have a single compromise solution, and the decision maker specifies his preferences concerning the objectives before solutions are calculated; thus, his preferred compromise solution is already specified. Some commonly used a priori methods are hierarchical optimization (also called lexicographical ordering), objective dominance, objective weighting, and goal programming.

With regard to the planning problem at hand, we use a MODM/MOP-based approach that integrates the lead-time minimization and resource leveling objectives into a single objective function. This objective function consists of two components (one for each of the objectives) that each measure deviations (similar to the goal programming approach) and normalize the deviations by adjustment factors (cf. Vieira and Favaretto, 2006). After normalization, the objective function combines the deviations additively and

introduces a weighting factor to represent the decision maker's preference between the two objectives (as performed by the objective weighting approach). We use this a priori method because selecting the most suitable solution from a large set of solutions is a difficult and time-consuming task (Korhonen, 2005, also comes to this conclusion). This problem is exacerbated by the difficulty of interpreting resource leveling objective values without visualization. In this context, the weighting factor incorporates some aspects of more interactive solution methods, as it enables the decision maker to evaluate several purposefully calculated MPSs, with minimal planning effort. This approach is also preferred by the responsible manager at the aerospace company.

To solve the resulting planning problem, solution methods that address resource leveling are analyzed. Concerning the application of a suitable solution method, Fink and Voß (2003) conclude that the effectiveness of heuristics vitally depends on three characteristics of the problem at hand: problem size, intended solution quality, and available computation time. Furthermore, they stress the need to transfer the findings from the literature to "decision support systems dealing with real world problems" (see Fink and Voß (2003)).

Only a few publications outline exact approaches for resource leveling, such as the works by Easa (1989) and Bandelloni et al. (1994). The former introduces a linear programming approach that minimizes the deviations from an average resource level. Bandelloni et al. (1994) propose a dynamic programming approach for their resource leveling problem that minimizes the squared deviation from the average resource utilization. The work of Rieck et al. (2012) provides a comprehensive overview of such methods. Gather et al. (2011) conclude that exact methods are only suitable for small problem instances with 10–15 tasks; therefore, there is a body of literature concerning heuristic approaches for resource leveling (further details can be found in the literature cited above). Here, the work of Ballestín et al. (2007) is worth mentioning. They propose a population-based iterated greedy algorithm in a make-to-order manufacturing environment that is capable of solving instances with up to 1000 tasks.

Because of the complexity of the planning problem at hand, we require a method that can solve virtually any size of problem as well as the ease of use. Thus, the metaheuristic variable neighborhood search (VNS) forms the basis of our proposed solution method. VNS is a metaheuristic that "exploits systematically the idea of neighborhood change, both in the descent to local minima and in the escape from the valleys which contain them" (Hansen and Mladenović, 2005). In contrast to other local-search-based methods, VNS explores neighborhoods of increasing distance from the current incumbent solution. VNS jumps from this solution to a new solution in the case of an improvement. The main advantage of this procedure is that if "many variables are already at their optimal value, [they] will be kept and used to obtain promising neighboring solutions" (Hansen and Mladenović, 2001). Furthermore, Amiri et al. (2010) conclude that VNS "is capable of escaping from the local optima by systematic changes of the neighborhood structures during the search process". VNS has shown promise in various applications (cf., e.g., Ribeiro et al., 2008, or Perez-Gonzalez and Framinan, 2010, as well as Hansen et al., 2010, for an overview). To the best of our knowledge, VNS has never been used to solve resource leveling or similar problems; therefore, this is a worthy exploration.

Furthermore, we provide the following contributions to the literature: a large-scale application case (as called for by Fink and Voß, 2003) stemming from an aerospace company, a new tailor-made construction heuristic combined with a randomized variable neighborhood descent procedure that is capable of solving large-scale problems, and a comparison to small instances solved by the DIscrete and Continuous OPTimizer (DICOPT) within the General Algebraic Modeling System (GAMS).

3. Problem definition and mathematical formulation

The central role of MP as defined in this paper is the temporal and factual coordination of planning tasks (of different business functions) in an HPPS. To determine a suitable MPS, all requirements, constraints, and interdependencies (inputs as well as desired outputs) with other planning tasks have to be analyzed in detail.

3.1. Problem analysis

At the time of planning, the "frozen" part of the rolling planning horizon (cf., e.g., Bredström et al. (2011), or Vargas and Metters (2011), on this topic) has to be fixed with respect to the replenishment times of critical materials. Therefore, the MPS has to be calculated long before assembly starts. In the case of the aerospace company, this period is at least 9 months.

The final assembly of the machines is organized in a serial assembly line with m assembly stages a_i ($i=1, 2,\ldots, m$). Because of the product and process characteristics discussed above, almost every assembly task is manually executed by the workforce of each stage; therefore, the workforce is the limiting resource. As planning occurs long before the actual assembly, the workforce level can be adjusted before assembly starts. For this reason, we do not represent a resource restriction as a constraint as usual; instead, resource utilization is considered in the objective function (cf. Section 3.2). This approach is used to increase planning flexibility. Workforce availability is insofar restricted, as it depends on a factory calendar that defines workdays and non-workdays (e.g., rest days, weekends, or factory vacation). As a consequence, a time-dependent resource availability vector indicates whether the workforce resource is available (on workdays) or not (on non-workdays). Thus, a set ω containing all period indices of workdays in the planning horizon can be specified, whereby all days and workdays are indexed by $t=1, 2, \ldots, T$. This set is identical for all assembly stages.

The number and type of (forecasted) orders representing the machines to be assembled within the planning horizon is specified by an order portfolio provided by the Sales Planning department. Forecasts are necessary because customer orders are not available at the time of planning. Table 1 presents such an order portfolio.

Table 1
Order portfolio and planning relevant data.

Profile	a_1		a_2		a_3		…	a_m		l_j^{NET}	Number of orders with this profile
	$l_{j\,1}^{NET}$	$r_{j\,1}$	$l_{j\,2}^{NET}$	$r_{j\,2}$	$l_{j\,3}^{NET}$	$r_{j\,3}$		$l_{j\,m}^{NET}$	$r_{j\,m}$		
A	4	1.2	3	1.5	4	1.0		5	1.2	18	17
B	7	1.1	5	1.4	17	1.2		15	1.1	50	34
…											

The order portfolio consists of a defined number n of planned orders o_j (with $j=1, 2, \ldots, n$). Each of these planned orders has a dedicated order profile representing a certain product type and a basic product specification. The order profile defines the net lead-time l_{ji}^{NET} (processing time) that is required to process the assembly of order j at assembly stage i and a workforce requirement factor r_{ji} per assembly stage i that is constant over time. Here, the net lead-time is given by a number of workdays; the total net lead-time of an order j is denoted by l_j^{NET}. The order profile also contains a special type of bill of materials that defines critical materials and components for each assembly stage. Because of the long replenishment lead-times these critical materials require special attention during the planning process to achieve short customer delivery times. As the calculated MPS determines the assembly start ($S_{ji} \in \omega$) and completion dates ($C_{ji} \in \omega$) of each order at each assembly stage, the procurement and/or production of these critical materials can be scheduled before tangible customer specifications are available. Therefore, the assembly start dates of an order specify material supply deadlines for each assembly stage.

Due to the rolling planning approach, there may be a concentration of orders with short net lead-times at the end of the planning horizon; these orders would be pushed backward from planning step to planning step. To avoid this problem, a start date interval for the first assembly stage is defined: $S_{j1} \in [fps^{PH}, lps^{PH}]$, $\forall j$. The first possible period for allocation fps^{PH} is defined by the first workday of the planning horizon, and the last possible start period for allocation lps^{PH} can be calculated using the end of the planning horizon T and the longest net lead-time of all orders: $lps^{PH}=T \ominus \max\{l_j^{NET} | \forall j\}$. For the sake of readability, we define the symbols $\oplus$ and $\ominus$ as operators that add or subtract a certain number of workdays to or from a given date and return a new date. This date is always a workday; if the number of workdays is equal to one, the new date corresponds to the next or previous workday.

Given this planning environment, the lead-time minimization and resource leveling objectives are addressed. With regard to the former, note that the factory calendar has a significant impact on the actual (gross) lead-time L_j^{ACT} of an order j, such that $L_j^{ACT} = C_{jm} - S_{j1} + 1$ and $L_{ji}^{ACT} = C_{ji} - S_{ji} + 1$. This effect of the factory calendar on the actual lead-time is also illustrated in the upper part of Fig. 1, while the resulting resource utilization is depicted in the lower part. The strong impact of the factory calendar on the actual order lead-times can be clearly observed in the case of the aerospace company considered: the resulting gross lead-time varies between 130% and 210% of the net lead-time. Because of this effect by the factory calendar, the primary lead-time minimization objective is supported using two constraints. The first constraint is the "non-preemptive" constraint (cf. Blazewicz et al., 2007; Pinedo, 2009); that is, interruptions within an assembly stage are not allowed except in the case of non-workdays. The second constraint is the "no-wait" constraint (id.), which prohibits assembly interruptions between two successive assembly stages. Consequently, the MPS is completely defined by the assembly start dates of the first stage (S_{j1}) of all orders and their corresponding profiles. These two additional constraints also affect the resource leveling objective. Changing start dates to minimize actual lead-times directly influences the resource utilization at all assembly stages and can lead to utilization peaks. These peaks undermine the resource leveling objective. For example, if order o_3 starts 1 day earlier, the actual lead-time would be reduced to 4 days ($L_{3\,3}^{ACT} = 4$), but a resource utilization peak would arise at assembly station a_1 (depicted by the black rectangle in Fig. 1). Generally, shorter lead-times can be achieved if orders start in a way such that they are completed in advance of factory vacations or maintenance periods. However, such a schedule would lead to an accumulation of start dates in certain parts of the planning horizon and, thus, to undesired resource utilization.

According to this discussion and the results presented in Section 5.3, there is a conflict between the two objectives that has to be solved. This resolution is achieved by the multi-criteria objective function presented in the following section.

3.2. Multi-criteria objective function

Based on the discussion in Section 2, the criteria used to measure lead-time and resource leveling performance are now described. The integration of both criteria into a single objective function is also discussed.

The lead-time minimization objective is represented by a lead-time deviation criterion that measures the deviation between the net lead-time l_j^{NET} and the actual (gross) lead-time L_j^{ACT} of an order:

$$L_j^{ACT} - l_j^{NET} \tag{1}$$

Concerning the objective of resource leveling as defined by the root of the sum of squared deviations from a desired value, the direct influence of the desired value on the resulting resource utilization can be observed. Based on this observation and the discussion of work ranges by Roca et al. (2008), we extend their "Full" work range concept by intervals I_i defined for each assembly stage i and calculate the desired utilization value $\overline{u}_i$ based on these intervals. An interval I_i is defined by the first possible allocation date (fpa_i), the last possible allocation date (lpa_i) of an assembly stage i, the set of workday indices in each interval (ω_i^I), and the number of workdays ($|\omega_i^I|$) in the interval:

$$fpa_i = \begin{cases} \text{first workday in the planning horizon,} & \text{if } i = 1 \\ fpa_1 \oplus \min\left\{ \sum_{i'=1}^{i-1} l_{ji'}^{NET} \,|\, \forall j \right\}, & \text{otherwise} \end{cases}$$

$$lpa_i = \begin{cases} \text{last workday in the planning horizon,} & \text{if } i = m \\ lpa_m \ominus \min\left\{ \sum_{i'=i+1}^{m} l_{ji'}^{NET} \,|\, \forall j \right\}, & \text{otherwise} \end{cases}$$

Therefore, the optimal resource utilization $\overline{u}_i$ of an assembly stage i is defined as the mean resource utilization in the corresponding interval.

$$\overline{u}_i = \sum_{j=1}^{n} l_{ji}^{NET} r_{ji} / |\omega_i^I| \quad \forall i \tag{2}$$

To smooth resource utilization, the root of the sum of squared deviations between the actual resource utilization $U_{i\,t}$ (4) in period t at assembly stage i and the optimal resource utilization $\overline{u}_i$ (2) at period t at assembly stage i is used as criterion (3). The squared deviation in the following formula is used to penalize large deviations and explicitly avoid utilization peaks.

$$\sqrt{\sum_{t \in \omega_i^I} (U_{it} - \overline{u}_i)^2} \tag{3}$$

here, the actual resource utilization $U_{i\,t}$ per period t and assembly stage i is defined by the number of orders that have to be processed in this period ($X_{i\,j\,t} = 1$ if $o_{i\,j}$ is processed in period t, 0 otherwise) and the corresponding resource utilization factor $r_{i\,j}$:

$$U_{it} = \sum_{j=1}^{n} X_{jit} r_{ji} \quad \forall i, t \tag{4}$$

As stated above, it is necessary to integrate the lead-time minimization (LT) and resource leveling (RL) objectives into a single objective function. Therefore, we examined a large number of adjustment components to normalize the objective values and to be able to directly represent the decision maker's preference by a weighting factor α. Here, it must be taken into account that the adjustment components have to be independent of a single

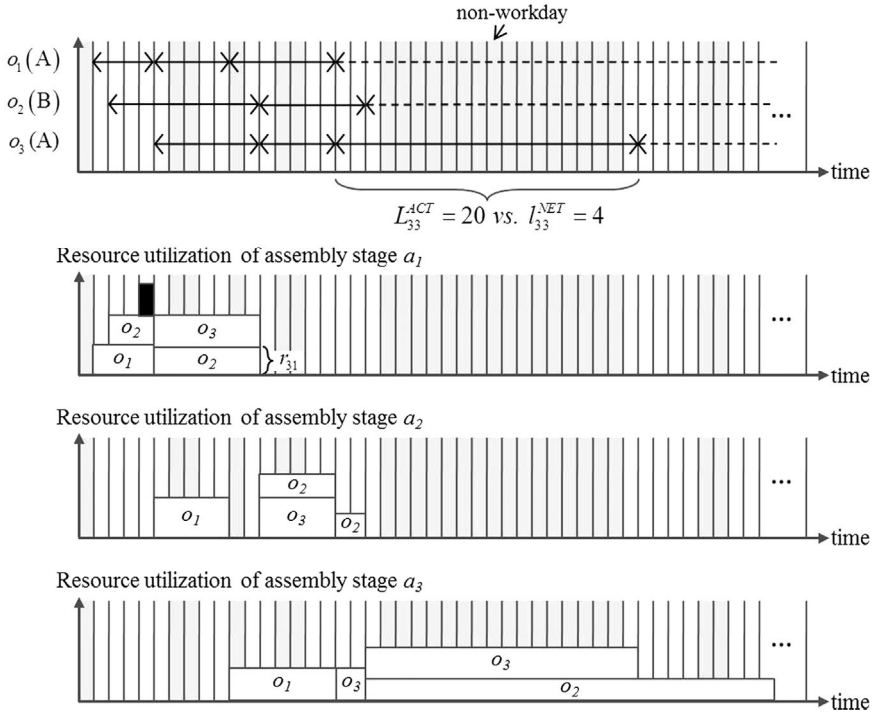

Fig. 1. MPS with three orders (two with profile A and one with profile B) and the resulting resource utilization.

problem instance. Thus, no manual update of the adjustment component is necessary. The result of these examinations is the following objective function.

$$\text{Min } \alpha \cdot \underbrace{\frac{1}{n}\sum_{j=1}^{n} \overbrace{\frac{1}{l_j^{NET}}}^{LT\ adjustment} \cdot (L_j^{ACT} - l_j^{NET})}_{LT} + (1-\alpha) \cdot \underbrace{\frac{1}{m}\sum_{i=1}^{m} \overbrace{\frac{1}{|\omega_i^I| \cdot \overline{u}_i}}^{RL\ adjustment} \cdot \sqrt{\sum_{t \in \omega_i^I} (U_{it} - \overline{u}_i)^2}}_{RL} \tag{5}$$

The results in Section 5.3 will show that the objective function developed in this section is able to address both objectives; that is, the objective function can minimize lead-times and achieve resource leveling independent of a single problem instance and with respect to the decision maker's preferences.

3.3. Mixed-integer non-linear formulation

To substantiate the constraints and objectives of the planning problem at hand and to solve small instances using DICOPT, we present a mixed-integer non-linear formulation as follows.Indices:

j	index for orders $\in \{1, ..., n\}$
i	index for assembly stages $\in \{1, ..., m\}$
t	index for days $\in \{1, \ldots, T\}$; including workdays and non-workdays

Parameter:

J	set of orders
I	set of assembly stages
T	length of the planning horizon
ω_i^I	set of workday indices in the possible allocation interval of assembly stage i; $\lvert\omega_i^I\rvert :=$ cardinality of ω_i^I
$\overline{u}_i$	desired resource utilization of assembly stage i
$l_{j\,i}^{NET}$	net lead-time (processing time) of order j on assembly stage i
l_j^{NET}	net lead-time of order j
lpa^{PH}	last possible start period at assembly stage one
fpa_i	first workday that can be allocated at assembly stage i
lpa_i	last workday that can be allocated at assembly stage i
$r_{j\,i}$	resource requirement factor of order j on assembly stage i
α	weighting factor

Variables:

$$X_{jit} = \begin{cases} 1, & \text{if order } j \text{ is processed on assembly stage } i \text{ at period } t \\ 0, & \text{otherwise} \end{cases}$$

$$S_{jit} = \begin{cases} 1, & \text{if order } j \text{ starts on assembly stage } i \text{ at the beginning of period } t \\ 0, & \text{otherwise} \end{cases}$$

$C_{ji} =$ index of the last period in which order o_j is processed on assembly stage i

Objective function:

$$\text{Min } \alpha \cdot \frac{1}{n}\sum_{j=1}^{n} \frac{\sum_{i=1}^{m}\sum_{t=fpa_i}^{lpa_i} X_{jit}\Big) - l_j^{NET}}{l_j^{NET}} + (1-\alpha) \cdot \frac{1}{m}\sum_{i=1}^{m} \frac{\sqrt{\sum_{t=fpa_i, t\in\omega_i^I}^{lpa_i} \left(\sum_{j=1}^{n} X_{jit} \cdot r_{ji}\right) - \overline{u}_i\Big)^2}}{|\omega_i^I| \cdot \overline{u}_i} \tag{6}$$

Constraints:

$$\sum_{t=1,\, t\in\omega_i^I}^{T} X_{jit} = l_{ji}^{NET} \quad \forall j, i \tag{7}$$

$$\sum_{t'=1}^{t} X_{jit'} \le \sum_{t''=1}^{t} S_{jit''} \cdot T \quad \forall j, i, t \tag{8}$$

$$\sum_{t=1}^{T} S_{jit} = 1 \quad \forall j, i \tag{9}$$

$$C_{ji} \geq X_{jit} \cdot t \quad \forall j, i, t \tag{10}$$

$$\sum_{t=1,\ t \in \omega_1^l}^{lpa^{PH}} S_{j1t} = 1 \quad \forall j \tag{11}$$

$$\sum_{t=1}^{T} X_{jit} = C_{ji} - \left(\sum_{t'=1}^{T} S_{jit'} \cdot t' \right) + 1 \quad \forall j, i \tag{12}$$

$$C_{ji-1} + 1 = \sum_{t=1}^{T} S_{jit} \cdot t \quad \forall j, i > 1 \tag{13}$$

$$X_{jit},\ S_{jit} \in \{0, 1\} \quad \forall j, i, t \tag{14}$$

$$C_{ji} \in \mathbb{Z}_{\geq 0} \quad \forall j, i \tag{15}$$

The first part of the combined multi-criteria objective function (6) is responsible for minimizing lead-times, while the second part addresses resource leveling. Constraint (7) ensures that the number of workdays that an order is processed at an assembly stage is equal to the number of workdays given by the net lead-time of this order at the corresponding assembly stage. Constraints (8) and (9) set the start date (the first allocation on an assembly stage) and constraint (10) sets the completion date of an order. The adherence of the start date interval of assembly stage one is assured by constraint (11). Constraint (12) prevents preemption at an assembly stage. Constraint (13) represents the no-wait condition between two consecutive stations and assures the proper sequence of assembly stages. Constraints (14) and (15) restrict the domains of the decision variables.

4. Solution method

Due to the complexity of the planning problem at hand and the requirement to solve virtually any size of problem, a solution method is required that is robust in terms of solution quality and efficient in terms of computation time. Therefore, we combine a problem-specific construction heuristic with a local search heuristic based on the principles of VNS.

4.1. Construction heuristic

The basic idea behind this problem-specific heuristic is to distribute all orders over the planning horizon equally. Hence, the uninformed construction heuristic is called equal distribution (EQD). The central requirement for the construction heuristic is that, based on the given input data (order portfolio, planning horizon, and factory calendar), all defined constraints must be respected to calculate feasible solutions. The inputs for the calculation of the initial solution are a list of order profiles (*opList[op, #]*) with the number of orders (#) of each order profile (*op*), the borders of the start day interval (fps^{PH}, lps^{PH}), and the total number of orders (n). EQD consists of two main steps. At first, n start dates with respect to the start date interval $[fps^{PH}, lps^{PH}]$ are calculated. Then, orders are assigned to the start dates according to their profile and the number of orders of each profile. The following pseudocode illustrates the main course of action of EQD:

```
EQD (opList[op, #], fps^PH, lps^PH, n)

// generate start dates
sdr←getWorkdays (fps^PH, lps^PH) / n      // calculation of a start date rhythm
S_1_1←fps^PH                              // set first start date
for j←2 to n do
    S_j_1←S_j−1_1 ⊕ roundSD (j, sdr)      // set next start date
    if S_j_1 > lps^PH then S_j_1←fps^PH   // if more orders than workdays in the planning horizon has to be planned
    end if;
next;

// assign order profiles
sortDesc (opList [op, #])                 // sort order profiles according to non-increasing numbers #
opList [op, #, q]←getOPR (opList[op, #])  // calculate order profile ratios q
assignOrders (opList [1, #, q])           // assigns all orders of the first profile to the earliest start dates
for z ←2 to total number of order profiles do
  for k ←1 to number of orders of profile z do
    insertOrders (k, roundOP (opList [z, #, q]))   // insert orders according to their ratio
  next;
next;
```

The first step of EQD is to generate start dates to which the orders will be assigned in the second step. For this purpose, a start date rhythm (*sdr*) is determined by dividing the number of work days in the start date interval (*getWorkdays*) by the total number of orders (n) that have to be planned. Then, beginning with the first possible start date, all further start dates are calculated based on the start date rhythm and a specific rounding function (*roundSD*). This function is required because the start date rhythm is not necessarily an integer value and standard rounding does not lead to suitable results. On the one hand, always rounding down entails an inappropriate start date distribution (not the full planning horizon is used). On the other hand, always rounding up would not guarantee that all orders start within the planning horizon. Therefore, the aim of this rounding function is to round up as often as possible to achieve, preferably, an equal distribution of start dates over the whole planning horizon.

In the second step of EQD, order profiles are assigned to the generated start dates. First, all order profiles are sorted according to non-increasing numbers of occurrence (*sortDesc*), and the ratio of occurrence times (q) of any order profile to the most frequent order profile is then calculated (*getOPR*). Thereafter, the orders of the most frequent order profile are assigned to the earliest unallocated start dates (*assignOrders*). This assignment results in a list of orders with start dates. Afterward, the remaining orders are inserted into this list iteratively, according to the order profile ratio (*insertOrders*). As these ratios are also not necessarily integer values, another specialized rounding function has to be applied (*roundOP*). The insertion of a new order into the list leads to a shifting of all orders currently starting behind the actual insertion position. By doing so, the different order profiles are distributed as equally as possible over the planning horizon. These insertions are executed until all orders of all profiles are assigned to start dates.

The time complexity of this deterministic heuristic depends on the total number of orders n. Assuming that the time complexity of the sorting algorithm is $O(n \log n)$, the time complexity of EQD is determined by the last two nested for-loops and the inserting procedure. Because inserting procedures have a complexity of $O(n)$ in most cases (depending on the data structure) and the insertion

is called $n-1$ times at most (within the nested for-loops), the time complexity of EQD is $O(n^2)$.

4.2. Improvement procedure

Generally, the metaheuristic VNS embeds a local search heuristic within a framework that uses changing neighborhood structures to explore local optima and escape from them (Hansen et al., 2010). The main aspects of the application of VNS to specific problems are – besides the local search procedure – the definition and number of appropriate neighborhood structures, the order in which these structures should be searched, and the strategy that should be used for changing neighborhoods (Hansen and Mladenović, 2001).

The first aspect is specifying transformation rules to obtain new neighboring solutions. These transformation rules define the basis for the local search procedure and the neighborhood structures. As stated above, an MPS is completely determined by start dates and assigned orders. Therefore, two possibilities to transform solutions exist: swap the assigned orders with different profiles or alter the start dates. Because many more combinations of start dates and orders can be reached by altering start dates, we use start date transformations to create neighboring solutions. Hence, the transformation rules define a sequence of integers that determines the number of workdays the start dates are shifted forward or backward in time. All constraints still must be satisfied to ensure that only feasible solutions are considered.

One possibility for building such a sequence of integers would be 1, 2, 3, 4, …, $T/2$. This sequence will be referred to as "*all*". Another possibility, aimed at reducing the number of solutions in a neighborhood and thus increasing the efficiency of the improvement procedure, is to use Fibonacci numbers to define the integer sequence (without the first two elements of the Fibonacci sequence). Following Zeckendorf's theorem, every natural number can be expressed as a sum of non-consecutive Fibonacci numbers (for a mathematical proof, see Lengyel (2006)). Therefore, this sequence provides a way to reach every other start date in the start date interval in a finite number of moves. Starting with the lowest Fibonacci number, all Fibonacci numbers embody two transformation rules: one from adding and the other from subtracting the Fibonacci number (in workdays) to or from a given start date. The largest Fibonacci number specifying a neighbor is the largest number that is smaller than half of the number of workdays in the start date interval. For example, if there are 50 workdays, the sequence is defined by 1, 2, 3, 5, 8, 13 and 21. To evaluate the two transformation sequences, the parameter *transRule* is set to "*all*" for the simple sequence and to "*fib*" for the Fibonacci numbers.

The neighborhood structures themselves are determined by the number of start dates that will be changed simultaneously. The first neighborhood structure performs separate start date shifts (according to the transformation sequence) to a given set of start dates. Thus, each start date is shifted and evaluated separately. The second structure performs the transformations on all combinations of two of the given start dates; the third structure, to all combinations of three start date; and so on. The last structure performs the transformation to all given start dates simultaneously. In this context, the parameter *searchDir* is used to control the sequence in which the structures are evaluated: either starting with the first structure (*searchDir* = "*asc*") or starting with the last structure (*searchDir* = "*desc*").

Because the number of start dates in an MPS corresponds to the number of orders in the order portfolio and this number could be large depending on the problem instance, the number and size of the neighborhood structures would increase rapidly. This increase in the number and size of the neighborhood structures would lead to a very time-consuming local search within a single neighborhood structure. For this reason, the number and size of structures is controlled by the parameter *selSD* that defines a certain number of start dates that will be randomly selected from all available start dates. For example, if *selSD* = 3, we obtain three structures: the first structure contains three elements with single start dates, the second structure contains three elements with two start dates each, and the third structure contains a single element with all three start dates. The local search procedure based on the transformation rules is executed separately for each of these elements. Depending on the improvement strategy (the manner in which better solutions will be selected), a new solution will become the incumbent solution. Generally, the strategies "best improvement/steepest decent" and "first improvement/first descent" can be applied. To compare both strategies, the parameter *impStrat* can be set to "*bi*" and "*fi*".

As we assume that the developed construction heuristic achieves a high-quality initial solution and as we seek to further reduce the computation time, we are not using an explicit shaking mechanism that accepts solutions that are worse than the incumbent one. As a result, and because of the randomly selected start dates, the proposed improvement procedure can be considered a randomized Variable Neighborhood Descent (RandVND) method. The main course of action of the procedure is described by the following pseudocode:

RandVND (*selSD, transRule, searchDir, impStrat, maxIt*$_{wi}$, *maxIt*$_{total}$)

```
// init
s*←EQD(…)                                   // get initial solution
trList←prepTR (transRule, getWorkdays())    // prepare
                                            transformation rules
// iterations
it_wi←0, it_total←0
l1: repeat
    sdList←getRandomSD (selSD, s*)          // select start dates
                                            randomly
    k←1
    l2: repeat
      // generate list of combinations
      combList←getCombinations (k,
      sdList, searchDir)
      // get new solution by local search
      s'←performLS (combList, trList,
      impStrat, s*)
      // set improved solution or change
      the neighborhood
      if getOV(s') < getOV(s*) then
        s*←s', k←1, it_wi←0
        exit l2;
      else
      ←k+1, it_wi←it_wi+1
      end if;
    until k=selSD ;
    it_total←it_total +1
until it_wi=maxIt_wi or it_total=maxIt_total ;
```

RandVND starts with the initial solution calculated by EQD and the preparation of the transformation rules (*prepTR*). The first step of the iterative part is the random selection of *selSD* start dates (*getRandomSD*) from the incumbent solution s^*. Within the inner loop (l2), the elements of the current neighborhood structure (k) are calculated (*getCombinations*) according to parameter *searchDir*. Next, the local search procedure is executed (*performLS*),

and depending on the improvement strategy (*impStrat*), a new solution s' will be returned. If s' improves the incumbent solution s*, s* will be replaced by s', and the next iteration of the outer loop (l1) will be executed. Otherwise, the next neighborhood structure will be explored. The procedure terminates whether either the maximum number of total iterations or the maximum number of iterations without improvement is reached.

The time complexity of an iteration of the inner loop of RandVND depends on the parameters *selSD*, *impStrat*, and *transRule*. In the worst case, *transRule* is set to "*all*" and has a length of $T/2$ and the improvement strategy is set to "*bi*" (all neighboring solutions have to be evaluated). In this case, the time complexity is given by $O(2^{selSD})$, whereby $selSD \ll T$.

5. Experiments and results

All of the subsequently described experiments are carried out within a comprehensive Microsoft® Excel spreadsheet tool, which contains EQD and RandVND (implemented in Visual Basic for Applications) and several modules for the preparation of the data, evaluation of the experiments, and visualization of the MPS and the resource utilization. This spreadsheet tool is also used by the aerospace company from which the planning problem originates.

To evaluate our solution method, we compare its performance with the commercial solver DICOPT (cf. Kocis and Grossmann, 1989). DICOPT is capable of solving mixed integer non-linear problems (MINLP) and is based on an extension of the outer-approximation algorithm for equality relaxation proposed by Varvarezos et al. (1992). The MINLP is decomposed into a continuous optimization sub-problem and a discrete optimization master problem that are solved separately. The former can be computed using any non-linear programming (NLP) solver, and the latter can be solved by applying any mixed integer programming (MIP) solver. We use CONOPT (introduced by Drud (1994)), a generalized reduced-gradient algorithm involving sparse non-linear constraints, for the NLP sub-problem and Gurobi for the MIP master problem. All solvers mentioned are part of GAMS.

5.1. Application case and test instances

The first group of instances (named "ac-instances") is based on anonymized real-world data from an aerospace company. The company is one of the world's leading companies in its sector and manufactures an average of 120 high-end aircraft per year. Each aircraft differs significantly in its functionality regarding communication, navigation, radar, and mission devices. The final assembly considered is organized as a serial production line with 10 stages: base unit and mechanical equipment, avionics systems, electronic systems, system integration, system tests, engine and final mechanical assembly, final electronic assembly, final avionics assembly, system completion, and final tests. The aircrafts, which are individually specified by the customers, can be categorized into three types, represented by three basic types of order profiles. The first type ("short profiles") represents so-called pre-assembly kits where only the first stages will be executed in the final assembly regarded here and the other stages will be executed elsewhere. Their net lead-times range between 12 and 18 workdays. The second type ("middle profiles") represents lightweight types with limited mission equipment. The third type is highly equipped ("long profiles"). The net lead-time of middle profiles ranges between 30 and 40 workdays, and the net lead-time of long profiles ranges between 50 and 60 workdays. These net lead-times have led to a mean gross lead-time of 104.79 days over the past few years. For every type, particular profiles have a "peak" lead-time on the fourth assembly stage. Concerning the aerospace company, the workforce requirement is identical for all profiles and assembly stages, and therefore, $r_{j\,i}$ is set to one for all orders and stages. To show the suitability of the planning method, the order profiles are combined to 12 test instances that differ in the order profile mix. The first and second application case instances (AC1, AC2) only include profiles of middle length. AC2 includes profiles with peaks. Instances AC3 and AC4 only consist of long profiles. AC4 contains profiles with peaks. Instances AC5, AC6, and AC7 include middle and long profiles. Instance AC6 includes only a few profiles with peaks, while AC7 includes many profiles with peaks. Instances AC8, AC9, and AC10 also contain short profiles; AC9 includes a few profiles with peaks, and AC10 includes many profiles with peaks. Instances AC11 and AC12 are completely random. Each of these scenarios is evaluated with 80, 120, and 200 orders of certain profile types. Further data inputs are the start and end of the planning horizon (365 days) and the factory calendar of the company. Further details on these instances and the order profiles can be found in the Appendix (Tables 7 and 8).

A second group of test instances (named "csb-instances") is used to establish the benchmark with the commercial solver. These instances are based on the first two basic profile types described above, but with reduced net lead-times (between 8 and 12 or 14 and 18 workdays), fewer assembly stages (5 or 8), and a reduced planning horizon (30 or 45 days) to be solvable within a reasonable amount of time using DICOPT (details about these instances also can be found in the Appendix; Tables 9 and 10). For this group of instances, the workforce requirement factors $r_{j\,i}$ are derived from the net lead-time. Here, $r_{j\,i}$ is directly proportional to the net lead-time (case A) or indirectly proportional (case B) and calculated as follows:

$$r_{ji}^{A} = \mathrm{Min}(1 + l_{ji}^{NET} \cdot 0.05,\ 2)$$

$$r_{ji}^{B} = \mathrm{Max}(2 - l_{ji}^{NET} \cdot 0.05,\ 1)$$

As a result, there are 16 small test instances (csb1A, csb1B, csb2A, csb2B, …, csb8B).

The third group of instances (named "ta-instances") is adapted from Taillard's flow shop sequencing instances (Taillard, 1993). Instances ta005, ta015, ta025, ta035, ta045, and ta055 are selected. The given processing times are adjusted by a factor of 0.1 to generate appropriate net lead-times. The planning horizon is defined as five times the mean net lead-time of all orders of an instance ($T = 5 \cdot \bar{l}_{j}^{NET}$). Again, the workforce requirements are set for each instance for the two cases A and B, resulting in 12 instances (ta005A, ta005B, ta014A, ta014B, …, ta055B).

5.2. Experiment and parameter settings

To evaluate the proposed solution method, three different groups of experiments based on the different problem instances are performed. The first group uses the csb-instances to evaluate the solutions obtained by RandVND compared to those obtained by DICOPT. In addition, the computing efficiency is examined while varying the flow control parameters *transRule*, *searchDir*, and *impStrat*. The second group of experiments, based on the ta-instances, is used to validate the results of the first group of experiments and to analyze the computing efficiency in more detail. The experiments in group two are all performed with $\alpha = 0.5$. Finally, the third group of experiments uses the ac-instances to show the suitability of the solution method for solving large problem instances efficiently, the independence of the adjustment components of the objective function from a specific problem instance, and the effectiveness of the weighting factor α in representing the decision maker's preferences.

As only the small csb-instances can be meaningfully compared using the objective value, two additional criteria are introduced. The first one addresses the objective of lead-time minimization and is based on the sum of deviations from the actual lead-time L_j^{ACT} and the minimal lead-time L_j^{BEST} that can be achieved if order j starts within $[fps^{PH}, lps^{PH}]$. L_j^{BEST} differs from l_j^{NET}, as it depends on the given factory calendar. Hence, a relative lower bound deviation ($rel.\ \Delta^{LT}$) for the lead-time objective can be calculated as follows:

$$rel.\Delta^{LT} = \left(\sum_{j=1}^{n} L_j^{ACT} - L_j^{BEST} \right) / \sum_{j=1}^{n} L_j^{ACT} \tag{16}$$

The second criterion is used to assess the resource leveling objective and measures the absolute deviation from the mean resource utilization:

$$\Delta^{RL} = \sum_{i=1}^{m} \sqrt{\sum_{t \in \omega_i^l} (U_{i\,t} - \overline{u}_i)^2} \tag{17}$$

Concerning the RandVND parameters *selSD*, $maxIt_{wi}$, and $maxIt_{total}$, a large number of tests have shown that they can be derived from the number of orders n in the following manner:

$$selSD = \begin{cases} 2, & \text{if } n < 50 \\ 4, & \text{otherwise} \end{cases},$$

$$maxIt_{wi} = \max\{n/3,\ 20\},$$

$$maxIt_{total} = \max\{15 \cdot n,\ 300\}$$

These values have been shown to be most efficient in terms of solution quality and computation time.

The solvers DICOPT, CONOPT, and Gurobi are parameterized using default values aside from the following settings (details of the parameters can be found in the solver documentation of GAMS):

Table 2
Parameter settings for DICOPT, CONOPT, and Gurobi.

DICOPT	reslim=36,000 seconds; maxcycles=50; stop=0; threads=0
CONOPT	continue= 1
Gurobi	optcr=0.01; mipreslim=3600 seconds

5.3. Computational results

Before discussing the results in detail, it should be noted that in this section all values calculated by RandVND are presented as mean values (each based on seven experimental runs), all values in columns with headings containing "rel." are presented as percentages, and computation times are presented in seconds (unless otherwise specified).

The results of the first group of experiments are summarized in Tables 3 and 4. The tables present the mean results for csb-instances csb1A to csb4B and csb5A to csb8B, respectively. Each table is separated into four groups of columns for different weighting factors. The third row contains the results from DICOPT: the objective value OV (calculated by (6)) and the two additional criteria $rel.\ \Delta^{LT}$ and Δ^{RL}. The following rows show the relative

Table 3
Mean results for csb-instances csb1A–csb4B.

α	0.1				0.5				0.9				1			
DICOPT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT
	0.4219	0.04	500.1	10079	0.3504	0.02	150.9	713	0.3706	0.00	195.6	404	0.3771	0.00	550.7	311
RandVND	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT
asc, FI, FIB	42.25	−4.99	78.16	4.0	−3.98	−2.46	12.40	4.3	−14.84	−3.37	29.79	3.5	−15.01	−3.23	75.96	3.4
asc, FI, ALL	42.25	−4.83	78.24	5.5	1.81	−0.10	9.98	6.9	−0.33	−0.10	33.47	7.8	−0.34	−0.07	72.76	7.3
asc, BI, FIB	38.52	−6.94	76.35	1.0	−19.48	−9.12	18.43	0.9	−39.30	−10.78	47.90	0.9	−45.57	−10.78	81.77	1.0
asc, BI, ALL	38.52	−6.94	76.35	1.3	−19.48	−9.12	18.43	1.4	−39.30	−10.78	47.90	1.0	−45.57	−10.78	81.77	1.3
desc, FI, FIB	42.66	−5.06	78.39	19.4	−0.85	−1.29	11.88	24.7	−4.63	−0.95	27.20	32.9	−5.59	−1.20	69.51	24.9
desc, FI, ALL	42.75	−4.88	78.51	32.9	1.93	0.01	9.42	44.0	−0.56	−0.01	27.96	40.1	−0.08	−0.02	70.32	39.0
desc, BI, FIB	42.69	−5.18	78.45	21.9	0.01	−1.05	12.31	30.2	−3.89	−0.74	24.57	33.7	−5.05	−1.09	70.31	26.2
desc, BI, ALL	42.81	−4.92	78.59	32.5	2.12	0.02	9.88	49.9	−0.68	0.00	27.08	39.7	0.00	0.00	70.20	37.9
RAND	31.92	−7.94	70.82		−25.46	−10.21	−1.66		−44.85	−12.10	34.16		−49.09	−11.72	77.17	

Table 4
Mean results for csb-instances csb5A to csb8B.

α	0.1				0.5				0.9				1			
DICOPT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT	OV	$rel.\ \Delta^{LT}$	Δ^{RL}	CT
	0.2538	8.12	111.26	36,001	0.3465	4.60	114.65	36,001	0.4229	0.61	268.73	36,002	0.4223	0.00	528.73	36,001
RandVND	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT	$rel.\ I^{OV}$	$rel.\ I^{\Delta LT}$	$rel.\ I^{\Delta RL}$	CT
asc, FI, FIB	22.46	0.23	3.24	12	4.24	−0.64	0.04	13	−0.60	−0.13	26.80	16	−1.54	−0.42	41.13	13
asc, FI, ALL	23.23	0.41	4.34	28	4.59	−0.64	1.73	23	−0.49	−0.30	32.34	28	−1.60	−0.45	41.35	25
asc, BI, FIB	8.97	−1.19	−17.10	1	−8.06	−4.70	−11.29	1	−23.22	−8.70	60.34	1	−32.01	−9.31	75.12	1
asc, BI, ALL	8.97	−1.19	−17.10	2	−8.06	−4.70	−11.29	2	−23.22	−8.70	60.34	2	−32.01	−9.31	75.12	2
desc, FI, FIB	22.86	0.15	3.78	80	4.26	−0.72	1.40	77	−0.92	0.14	20.15	137	−1.29	−0.35	36.17	86
desc, FI, ALL	23.50	0.14	4.92	128	4.60	−0.62	2.03	153	−0.78	−0.28	29.42	181	−1.29	−0.35	37.51	165
desc, BI, FIB	22.81	0.22	3.77	72	4.26	−0.61	0.06	70	−0.75	0.10	22.89	110	−1.29	−0.35	37.66	88
desc, BI, ALL	23.38	0.32	4.48	122	4.69	−0.61	2.29	145	−0.65	−0.24	29.96	204	−1.33	−0.37	38.72	167
RAND	−0.22	−1.51	−41.00		−12.45	−4.98	−31.84		−25.08	−8.97	53.56		−33.66	−9.62	70.89	

Table 5
Results of the ta-instances with *transRule*="*all*" and *transRule*="*fib*".

Inst.	n	m	RandVND ("asc", "fi")							
			"fib"				"all"			
			OV	rel. Δ^{LT}	Δ^{RL}	CT	OV	rel. Δ^{LT}	Δ^{RL}	CT
ta005A	20	5	0.6812	13.05	46.70	47	0.6894	12.87	48.13	157
ta005B	20	5	0.6951	13.75	63.86	42	0.6994	13.34	65.69	230
ta015A	20	10	0.7434	15.50	95.04	192	0.7338	14.51	94.82	1012
ta015B	20	10	0.7577	16.03	131.73	149	0.7416	15.41	128.63	576
ta025A	20	20	0.7411	7.67	221.99	320	0.7370	6.86	223.13	5772
ta025B	20	20	0.7498	8.05	298.08	297	0.7350	6.75	295.53	3814
ta035A	50	5	0.6832	12.48	123.41	107	0.6632	12.04	118.75	338
ta035B	50	5	0.6948	13.16	165.22	69	0.6665	12.31	156.75	576
ta045A	50	10	0.6133	16.78	176.72	226	0.6003	15.91	173.28	2384
ta045B	50	10	0.6107	16.98	232.01	210	0.6044	15.77	235.84	1769
ta055A	50	20	0.5920	8.14	351.86	1159	0.5828	7.77	345.97	12,876
ta055B	50	20	0.5947	7.98	477.41	1503	0.5855	8.10	460.77	10,060

improvements *rel.* I^{OV}, *rel.* $I^{\Delta LT}$, and *rel.* $I^{\Delta RL}$ of RandVND with different settings for *searchDir*, *impStrat*, and *transRule* and the randomly generated solutions (RAND; 100 solutions per instance) compared to the solutions calculated by DICOPT. The last column in each group depicts the mean computation times (CT) Table 2.

The results presented in Tables 3 and 4 show that the proposed solution method (neglecting settings "asc", "bi", "fib" and "asc", "bi", "all") is superior compared to the results achieved by random sampling. Due to their poor performance, the two settings mentioned in the parenthetical above are also neglected in the following analysis.

The results produced by DICOPT are questionable in terms of achieving the resource leveling objective when $\alpha=0.1$ and for csb-instances csb1A–csb4B (cf. Table 3). In contrast, if $\alpha=1$, then DICOPT is able to achieve the best possible lead-time L_j^{BEST} (*rel.* $\Delta^{LT}=0$). Altogether, DICOPT achieves its best results with $\alpha=0.5$ and is able to solve the MPS problem with the non-linear objective function in a suitable manner. Table 4 shows that DICOPT behaves as expected (with regard to the different weighting factors) and that it always reaches the defined computation time limit.

The results suggest that RandVND outperforms DICOPT with respect to the resource leveling objective (*rel.* $I^{\Delta RL}$ between 9.42% and 78.59% in Table 3 and between 0.04% and 41.35% in Table 4). In addition, RandVND performs sufficiently well in computing the smaller instances csb1A to csb4B (*rel.* $I^{\Delta LT}$ between −5.18% and 0.02% in Table 3) and very well in computing instances csb5A to csb8B (*rel.* $I^{\Delta LT}$ between −0.72% and 0.41% in Table 4). Regarding the last observation, the computation time performance of RandVND is analyzed in more detail, as it is of special importance: the csb-instances are very small, and the computation time increases disproportionally with the size of the problem instance. Parameters *searchDir*="*asc*", *impStrat*="*fi*" and *transRule*="*fib*" perform comparatively well considering the improvement of the relative deviation of the objective value (*rel.* I^{OV}) and the two other criteria; however, the performance is especially impressive in terms of the computation time. The benefit of using transformation rules based on Fibonacci numbers can be observed when comparing the computation times. The solution quality for criteria *rel.* I^{OV}, *rel.* $I^{\Delta LT}$, and *rel.* $I^{\Delta RL}$ are almost identical, but the computation times are significantly shorter compared to the simple integer sequence.

Because the search direction "*asc*" outperforms "*desc*" in terms of the computation time, the following results focus on the difference between the two transformation rule integer sequences.

The results in Table 5 show that the computation time advantages of the Fibonacci integer sequence increase significantly with the size of the problem instance. A comparison of the objective values in Table 5 across the different resource requirement calculation rules shows that nearly the same objective value is achieved in both cases (A and B). This result suggests that the adjustment factors are suitable for different problem instances.

Table 6
Mean computation times (in min).

n/α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
80	41	48	48	39	45	49	60	60	85
120	85	84	79	77	85	93	105	102	144
200	138	154	158	157	171	193	198	224	313

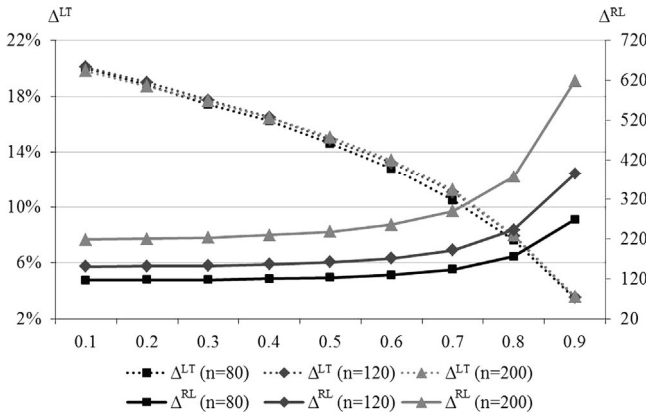

Fig. 2. Lead-time and resource leveling performance depending on α (for ac-instances).

The computation times (on an Intel® Core™ i7-2600 CPU @ 3.40 GHz with 16 GB RAM) presented in Table 6 highlight the importance of the computation time criterion for selecting the most suitable RandVND settings.

These computation times seem to be adequately short, especially concerning the level of planning, frequency of planning, and size of the ac-instances.

The lead-time and resource leveling performance as a function of α is illustrated in Fig. 2. The opposite trends of the curves for the lower bound criteria clearly show the effectiveness of α in controlling the tradeoff between the two objectives; thus, α is a suitable representation of the decision maker's preferences. The figure also visualizes the conflict between the two objectives. Focusing on the objective of lead-time minimization would lead to

highly variable resource utilization; focusing on resource leveling would greatly stifle the potential to reduce lead-times and, thus, reduce capital commitments.

Concerning the primary objective of lead-time minimization, it should be emphasized that with $\alpha=0.9$, the relative deviation in lead-time (*rel.* Δ^{LT}) can, on average, be reduced to 3.50% for all ac-instances. In addition, the adjustment components of the objective function work properly for all ac-instances.

Moreover, Fig. 2 provides the following non-predictable insights. In the interval from $\alpha=0.1$ to $\alpha=0.5$, the lead-time performance increases, while the resource leveling performance is almost constant. In contrast, when α is greater than 0.8, both performance criteria behave as expected. Consequently, the most reasonable interval ranges from $\alpha=0.5$ to $\alpha=0.8$. Here, the resource leveling performance only decreases slightly with increasing α, while the lead-time performance increases significantly. This observation leads to the following implications for management. Within this interval, the decision maker can define an α-value that adequately reflects his preferences. If the decision maker is not able to specify his preferences a priori, a small number of MPS could be calculated (with α-values within this interval), and the most suitable one could be selected. This interval provides a rule of thumb for setting the initial α-value.

6. Conclusions

The general purpose of the multi-criteria MPS approach for special purpose machinery presented is the temporal and factual coordination of all dependent planning tasks to achieve the basic goals cost reduction and customer satisfaction. In this context, capital commitments, contractual penalties, and compensation costs are of special interest. To minimize these costs, the objective function combines lead-time minimization and resource leveling objectives, and a weighting factor is used to balance the objectives with respect to the decision maker's preferences. This objective function combined with the underlying production system leads to a new optimization problem that needs to be solved not only for small problems but also for large real world problems. In this case, the small problem sizes are solved by a commercial solver. However, a tailor-made construction heuristic and a RandVND procedure are developed to efficiently solve problems of virtually all sizes. The computational results show that the implemented solution method is able to solve the small instances with an improvement of between −15.01% and 42.81% compared with the objective values obtained by the commercial solver DICOPT. Even large-scale problems (with 200 orders, 10 assembly stages, and a planning horizon of 1 year) can be solved with a solution of sufficient quality concerning both the lead-time and the resource

Table 7
Number of orders per order profile and scenario (ac-instances).

	n	Order profile												
		PT-S1	PT-S2-P	PT-S3	PT-S4-P	PT-M1	PT-M2	PT-M3-P	PT-M4	PT-M5	PT-M6-P	PT-L1	PT-L2-P	PT-L3	PT-L4-P
AC1	80					20	20		20	20					
	120					30	30		30	30					
	200					50	50		50	50					
AC2	80					13	13	14	13	13	14				
	120					19	19	22	19	19	22				
	200					32	32	36	32	32	36				
AC3	80											40		40	
	120											60		60	
	200											100		100	
AC4	80											20	20	20	20
	120											30	30	30	30
	200											50	50	50	50
AC5	80					13	13		13	13		14		14	
	120					20	20		20	20		20		20	
	200					32	32		32	32		36		36	
AC6	80					11	11	3	11	11	3	12	3	12	3
	120					16	16	6	16	16	6	16	6	16	6
	200					25	25	12	25	25	12	26	12	26	12
AC7	80					8	8	8	8	8	8	8	8	8	8
	120					12	12	12	12	12	12	12	12	12	12
	200					20	20	20	20	20	20	20	20	20	20
AC8	80	10		10		10	10		10	10		10		10	
	120	15		15		15	15		15	15		15		15	
	200	25		25		25	25		25	25		25		25	
AC9	80	9	2	9	2	8	8	2	8	8	2	9	2	9	2
	120	12	4	12	4	12	12	4	12	12	4	12	4	12	4
	200	20	7	20	7	19	19	8	19	19	8	20	7	20	7
AC10	80	4	4	4	4	8	8	8	8	8	8	4	4	4	4
	120	6	6	6	6	12	12	12	12	12	12	6	6	6	6
	200	10	10	10	10	20	20	20	20	20	20	10	10	10	10
AC11	80	4	11	7		5	4	3	2	3	6	9	9	12	5
	120	10	15	8	2	12	6	3	13		7	5	12	15	12
	200	22	24	16	21	3	10	18	9	2	23	6	7	16	23
AC12	80	4	2		4	8	6	2	7	5	5	12	4	9	12
	120	1	3	7	12	16	13	8	4	14	7	16	12	7	0
	200	16	17	15	20	21	19	16	6	12	16	8	1	11	22

Table 8
Net lead-times per order profile and assembly stage (ac-instances).

Order profile	l_j^{NET}	Assembly stage									
		a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
PT-S1	12	1	3	3	3	2					
PT-S2-P	12	1	2	2	5	2					
PT-S3	18	1	3	4	5	3	2				
PT-S4-P	18	1	3	3	7	3	1				
PT-M1	30	1	3	4	6	6	5	4	1		
PT-M2	32	2	5	5	5	5	5	3	2		
PT-M3-P	32	1	4	5	9	5	4	3	1		
PT-M4	38	2	4	5	6	6	5	4	3	2	1
PT-M5	40	2	3	6	7	6	4	5	4	2	1
PT-M6-P	40	2	3	5	12	5	3	3	3	2	2
PT-L1	50	3	4	6	7	7	7	6	5	3	2
PT-L2-P	50	2	4	4	10	6	6	6	6	4	2
PT-L3	60	3	6	8	8	7	8	6	7	4	3
PT-L4-P	60	3	5	6	13	7	6	6	7	4	3

Table 9
Number of orders per order profile and scenario (csb-instances).

Instance	n	m	Order profile					T
			S1	S2-P	M1	M2	M3-P	
csb1	10	5	5	5				30
csb2	15	5	7	8				30
csb3	20	5	10	10				30
csb4	30	5	15	15				30
csb5	10	8			4	3	3	48
csb6	15	8			4	4	7	48
csb7	20	8	4	4	4	4	4	48
csb8	30	8	4	4	7	7	8	48

Table 10
Net lead-times per order profile and assembly stage (csb-instances).

Order profile	l_j^{NET}	Assembly stage							
		a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
S1	12	1	3	3	3	2			
S2-P	12	1	2	2	5	2			
M1	30	1	3	4	6	6	5	4	1
M2	32	2	5	5	5	5	5	3	2
M3-P	32	1	4	5	9	5	4	3	1

leveling criterion within a reasonable amount of time. Moreover, the objective function and adjustment components are suitable for very different problem instances. The results also reveal that the α-value can adequately reflect the decision maker's preferences. Furthermore, we provide a rule of thumb for selecting an appropriate initial weighting factor.

The application of the MPS approach by the aerospace company has led to promising results. The developed Excel tool supports the ability of responsible decision makers to express and visualize their preferences and to balance the two objectives. The operational use of the planning approach and its solution methods has also shown that the results are robust to varying input data.

Further research should address the improvement of the proposed solution method and its competitiveness with other methods, such as Simulated Annealing or Tabu Search. Another research direction would be to analyze the effects of the resulting temporal framework and resource utilization on dependent planning tasks.

Appendix

See Appendix Tables 7–10.

References

Amiri, M., Zandieh, M., Yazdani, M., Bagheri, A., 2010. A variable neighbourhood search algorithm for the flexible job-shop scheduling problem. Int. J. Prod. Res. 48 (19), 5671–5689.

Anagnostopoulos, K.P., Koulinas, G.K., 2010. A simulated annealing hyperheuristic for construction resource levelling. Constr. Manage. Econ. 28 (2), 163–175.

Ballestín, F., Schwindt, C., Zimmermann, J., 2007. Resource leveling in make-to-order production: modeling and heuristic solution method. Int. J. Oper. Res. 4 (1), 50–62.

Bandelloni, M., Tucci, M., Rinaldi, R., 1994. Optimal resource leveling using non-serial dynamic programming. Eur. J. Oper. Res. 78 (2), 162–177.

Belton, V., Stewart, T.J., 2002. Multiple Criteria Decision Analysis: An Integrated Approach. Kluwer Academic Publishers, Boston.

Blazewicz, J., Ecker, K.H., Pesch, E., Schmidt, G., Weglarz, J., 2007. Handbook on Scheduling: From Theory to Applications (International Handbook on Information Systems). Springer, Berlin, Heidelberg, New York.

Bredström, D., Flisberg, P., Rönnqvist, M., 2011. A new method for robustness in rolling horizon planning. Int. J. Prod. Econ. (in press) doi:10.1016/j.ijpe.2011.02.008, 2011.

Caramia, M., Dell'Olmo, P., 2006. Effective Resource Management in Manufacturing Systems: Optimization Algorithms for Production Planning. Springer, London.

Drexl, A., Fleischmann, B., Günther, H.-O., Stadtler, H., Tempelmeier, H., 1994. Konzeptionelle Grundlagen kapazitätsorientierter PPS-Systeme. Z. Betriebswirtsch. Forsch. 46 (12), 1022–1045.

Drótos, M., Kis, T., 2011. Resource leveling in a machine environment. Eur. J. Oper. Res. 212 (1), 12–21.

Drud, A.S., 1994. CONOPT – a large-scale GRG code. INFORMS J. Comput. 6 (2), 207–216.

Dyer, J.S., 2005. MAUT – multiattribute utility theory. In: Figueira, J., Greco, S., Ehrgott, M. (Eds.), Multiple Criteria Decision Analysis: State of the Art Surveys. Springer Science+Business Media, Inc., New York, pp. 265–296.

Dyer, J.S., Fishburn, P.C., Steuer, R.E., Wallenius, J., Zionts, S., 1992. Multiple criteria decision making, multiattribute utility theory: the next ten years. Manage. Sci. 38 (5), 645.

Easa, S.M., 1989. Resource leveling in construction by optimization. J. Constr. Eng. Manage. 115 (2), 302–316.

Ehrgott, M., Gandibleux, X., 2000. A survey and annotated bibliography of multiobjective combinatorial optimization. OR Spektrum 22 (4), 425–460.

Figueira, J., Greco, S., Ehrgott, M. (Eds.), 2005. Multiple Criteria Decision Analysis: State of the Art Surveys. Springer Science+Business Media, Inc., New York.

Fink, A., Voß, S., 2003. Solving the continuous flow-shop scheduling problem by metaheuristics: meta-heuristics in combinatorial optimization. Eur. J. Oper. Res. 151 (2), 400–414.

Franck, B., Neumann, K., Schwindt, C., 1997. A capacity-oriented hierarchical approach to single-item and small-batch production planning using project-scheduling methods. OR Spektrum 19 (2), 77–85.

Franck, B., Neumann, K., Schwindt, C., 2001. Project scheduling with calendars. OR-Spektrum 23 (3), 325–334.

Gather, T., Zimmermann, J., Bartels, J.-H., 2011. Exact methods for the resource levelling problem. J. Scheduling 14 (6), 557–569.

Guitouni, A., Martel, J.-M., 1998. Tentative guidelines to help choosing an appropriate MCDA method. Eur. J. Oper. Res. 109, 501–521.

Gupta, Y.P., Evans, G.W., Gupta, M.C., 1991. A review of multi-criterion approaches to FMS scheduling problems. Int. J. Prod. Econ. 22 (1), 13–31.

Hans, E.W., Herroelen, W., Leus, R., Wullink, G., 2007. A hierarchical approach to multi-project planning under uncertainty. Omega 35 (5), 563–577.

Hansen, P., Mladenović, N., 2001. Variable neighborhood search: principles and applications. Eur. J. Oper. Res. 130 (3), 449–467.

Hansen, P., Mladenović, N., 2005. Variable neighborhood search. In: Burke, E.K., Kendall, G. (Eds.), Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques. Springer Science+Business Media LLC, Boston, MA, pp. 211–238.

Hansen, P., Mladenovic, N., Moreno Perez, J.A., 2010. Variable neighbourhood search: methods and applications. Ann. Oper. Res. 175 (1), 367–407.

Hoogeveen, H., 2005. Multicriteria scheduling. Eur. J. Oper. Res. 167 (3), 592–623.

Keeney, R.L., Meyer, R.F., Raiffa, H., 1993. Decisions with Multiple Objectives: Preferences and Value Tradeoffs. Cambridge University Press, Cambridge.

Kirkwood, C.W., 1997. Strategic Decision Making: Multiobjective Decision Analysis with Spreadsheets. Thompson, Belmont, California.

Kocis, G., Grossmann, I., 1989. Computational experience with dicopt solving MINLP problems in process systems engineering. Comput. Chem. Eng. 13 (3), 307–315.

Korhonen, P., 2005. Interactive methods. In: Figueira, J., Greco, S., Ehrgott, M. (Eds.), Multiple Criteria Decision Analysis: State of the Art Surveys. Springer Science+Business Media, Inc., New York, pp. 641–665.

Lengyel, T., 2006. A counting based proof of the generalized Zeckendorf's theorem. Fibonacci Quarterly 44 (4), 324–325.

Neumann, K., Schwindt, C., Zimmermann, J., 2003. Project scheduling with time windows and scarce resources: Temporal and resource-constrained project scheduling with regular and nonregular objective functions, 2nd ed.. Springer, Berlin, New York.

Neumann, K., Zimmermann, J., 1999. Resource levelling for projects with schedule-dependent time windows. Eur. J. Oper. Res. 117 (3), 591–605.
Perez-Gonzalez, P., Framinan, J.M., 2010. Setting a common due date in a constrained flowshop: a variable neighbourhood search approach. Comput. Oper. Res. 37 (10), 1740–1748.
Pinedo, M.L., 2009. Planning and Scheduling in Manufacturing and Services, 2nd ed.. Springer, Dordrecht, Heidelberg, London, New York.
Rhode, J., Wagner, M., 2008. Master planning. In: Stadtler, H., Kilger, C. (Eds.), Supply Chain Management and Advanced Planning: Concepts, Models, Software and Case Studies, 4th ed. Springer, Berlin, Heidelberg, pp. 161–179.
Ribeiro, C.C., Aloise, D., Noronha, T.F., Rocha, C., Urrutia, S., 2008. A hybrid heuristic for a multi-objective real-life car sequencing problem with painting and assembly line constraints. Eur. J. Oper. Res. 191 (3), 981–992.
Rieck, J., Zimmermann, J., Gather, T., 2012. Mixed-integer linear programming for resource leveling problems. Eur. J. Oper. Res. 221 (1), 27–37.
Roca, J., Pugnaghi, E., Libert, G., 2008. Solving an extended resource leveling problem with multiobjective evolutionary algorithms. Int. J. Comput. Intell. 4, 289–300.
Schwindt, C., Trautmann, N., 2000. Batch scheduling in process industries: an application of resource–constrained project scheduling. OR-Spektrum 22 (4), 501–524.
Taillard, E., 1993. Benchmarks for basic scheduling problems. Eur. J. Oper. Res. 64 (2), 278–285.
T'kindt, V., Billaut, J.-C., 2006. Multicriteria Scheduling: Theory, Models and Algorithms, 2nd ed.. Springer, Berlin, Heidelberg.
Trautmann, N., 2001. Calendars in Project Scheduling. In: Fleischmann, B., Lasch, R., Derigs, U., Domschke, W., Rieder, U. (Eds.), Operations Research Proceedings, vol. 2000. Operations Research Proceedings): Springer, Berlin Heidelberg, pp. 388–392.
Vargas, V., Metters, R., 2011. A master production scheduling procedure for stochastic demand and rolling planning horizons. Int. J. Prod. Econ. 132 (2), 296–302.
Varvarezos, D.K., Grossmann, I.E., Biegler, L.T., 1992. An outer-approximation method for multiperiod design optimization. Ind. Eng. Chem. Res. 31 (6), 1466–1477.
Vieira, G.E., Favaretto, F., 2006. A new and practical heuristic for master production scheduling creation. Int. J. Prod. Res. 44 (18), 3607–3625.
Vollmann, T.E., Berry, W.L., Whybark, D.C., Jacobs, F.R., 2005. Manufacturing Planning and Control Systems for Supply Chain Management, 5th ed.. McGraw-Hill, New York, Chicago, San Francisco.
Younis, M., Saad, B., 1996. Optimal resource leveling of multi-resource projects. Comput. Ind. Eng. 31 (1–2), 1–4.
Zhan, J., 1992. Calendarization of time planning in MPM networks. Z für Oper. Res. l (5), 423–438.