Bi-objective optimization of the tactical allocation of job types to machines: mathematical modeling, theoretical analysis, and numerical tests

Citation for the original published paper (version of record):
Fotedar, S., Strömberg, A., Almgren, T. (2023). Bi-objective optimization of the tactical allocation of job types to machines: mathematical
modeling, theoretical analysis, and numerical tests. International Transactions in Operational Research, 30: 3479-3507. http://dx.doi.org/10.1111/itor.13180

(article starts on next page)

INTERNATIONAL TRANSACTIONS
IN OPERATIONAL RESEARCH

WILEY

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 0 (2022) 1–29
DOI: 10.1111/itor.13180

Bi-objective optimization of the tactical allocation of job types to machines: mathematical modeling, theoretical analysis, and numerical tests

Sunney Fotedar[a,*], Ann-Brith Strömberg[a] and Torgny Almgren[b]

[a] *Department of Mathematical Sciences, Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden*
[b] *GKN Aerospace Sweden AB, Trollhättan, Sweden*
E-mail: sunney@chalmers.se [Fotedar]; anstr@chalmers.se [Strömberg]; torgny.almgren@gknaerospace.com [Almgren]

Received 16 October 2021; received in revised form 6 May 2022; accepted 10 July 2022

Abstract

We introduce a *tactical resource allocation* model for a large aerospace engine system manufacturer aimed at long-term production planning. Our model identifies the *routings* a product takes through the factory, and which machines should be qualified for a balanced resource loading, to reduce product lead times. We prove some important mathematical properties of the model that are used to develop a heuristic providing a good initial feasible solution. We propose a tailored approach for our class of problems combining two well-known criterion space search algorithms, the bi-directional ϵ-constraint method and the augmented weighted Tchebycheff method. A computational investigation comparing solution times for several solution methods is presented for 60 numerical instances.

Keywords: bi-objective mixed integer programming (BOMIP); production planning; aerospace industry; capacity planning

1. Introduction

Allocating resources among competing activities is one of the most popular type of optimization problems considered in various application areas, such as manufacturing, hospitals, and finance. Manufacturing, Planning, and Control (MPC) is a subject of interest to many practitioners as well as operations research experts due to its positive impact on efficiency for many companies globally. Some of the traditional MPC systems are integrated MPS (Master Production Schedule; Blackstone Jr., 2013) and MRP (Materials Requirement Planning; Plossl and Orlicky, 1999) models

*Corresponding author.

that optimize simultaneously the production and the purchase of raw materials over a shorter time horizon, taking the allocation of products to resources as fixed. It is well understood that having more options of verified resources could be useful for the planners to tackle any demand variations in the short term. In some industries, there is, however, a need to perform time-consuming or costly verification before re-allocating a product to a new resource. Hence, it is beneficial to identify such verifications (we use the term *qualifications*) before they are required.

Our proposed model (for case company GKN Aerospace, a leading manufacturer of aerospace engine systems) identifies the routes that a product may take through the factory over a fairly long time horizon (a quarter of a year in this work). The need of our allocation model is due to the requirement of costly as well as time and resource demanding qualification of production processes when new products are introduced or significant changes arise in the machining capacity (in this work, resources implies machines). The model considers multiple products, each having different part types that require machining operations (e.g., milling, turning, and grinding), and the capacity (in hours) of machines as well as the time horizon are considered to be finite. There is a tendency (at GKN) to utilize only a few capable machines (with respect to speed, tolerances, and multiple functionality) at high *loading levels* (Blackstone Jr., 2013, p. 94). This causes a *resource loading imbalance* in the production system that leads to variations in the delivery times of products, resulting in long queues of *jobs* (operations done on products/parts) in front of some machines. We relieve imbalance by identifying multiple routes by means of *qualifying* alternative machines for a given type of job. In summary, our model allocates jobs (operations performed on products/parts) to machines with the bi-objective goal to minimize both qualification cost and loading imbalance, while demand in each time period and various other side constraints are satisfied.

1.1. *Qualifications of machines for jobs*

A qualification of a machine for a job type involves verifying that all encoded tasks can be performed by the machine within acceptable tolerances. It involves writing a computer program for the control system, buying new fixtures, training the machining staff, and performing simulations to validate expected output. Once a machine is qualified for a given job type, it can be used for orders of the same job type in all subsequent time periods; hence, the associated costs are one-time costs, as opposed to set-up/start-up (changeover) costs commonly considered in *capacitated lot-sizing models* (Pochet and Wolsey, 2006, p. 137).[1] Table 1 illustrates the routings in a dummy production system with three machines and the two operations milling and turning, performed on a single product/part. For simplicity, we refer to *job type* as the combination of a product/part type and the operation performed on it. In the first time period, milling is done in machines 1 and 3, in the second time period, the same operation is done in machines 1 and 2, and in the third time period, it is done only in machine 2. For machine 2, the box around the M indicates that the milling operation is to be qualified and performed in time period 2. For machine 3, the turning operation is to be qualified in time period 2 and performed in time period 3. The blue and green colors indicate the time periods when a given machine is qualified to

[1]As the volumes are quite low in the aerospace industry, jobs are not performed in batches. Hence, set-up times are included in the processing times of jobs in machines.

Table 1
Routings for a single part/product: M (milling) and T (turning) indicate time periods (t) when machines (k) are used for the respective purpose; □ indicates time period and machine qualification for milling and turning, respectively

	$k = 1$		$k = 2$		$k = 3$	
$t = 1$	M	T		T	M	
$t = 2$	M	T	$\boxed{\text{M}}$	T		□
$t = 3$			M	T		T

perform milling and turning, respectively. As a few machines are pre-qualified (used previously) for some job types, not all machines need to be qualified. For example, machine 1 is pre-qualified for both M and T.

1.2. Contribution

The contribution of this work is twofold. We introduce a *tactical resource allocation problem* (TRAP) with unrelated parallel machines (with several company related side constraints and assumptions), and a corresponding mathematical model. We consider the trade-off between resource loading imbalance and qualification costs, which is to the best of our knowledge not considered in the literature. Some mathematical properties of the model are utilized in a starting heuristic for reducing the computing time. We then suggest a tailored algorithm for the TRAP, starting with a bi-directional ϵ-constraint method, followed by the Augmented weighted Tchebycheff (AWT) method. The approach tackles some well-known shortcomings of the ϵ-constraint method and is shown to shorten computing times, in particular for some difficult numerical instances. Our model can be used by most organizations where resources are shared among products/parts/customers and costly and/or time-demanding preparations are required the first time a product/part/customer needs to be (re)allocated to a new resource.

1.3. Outline

The scope of the TRAP is described in Section 2.1 along with some background, while Section 2.2 provides a brief introduction to multi-objective optimization methods. Section 3.1 presents a mathematical model for the TRAP; it is proven to be NP-hard in Section 3.2. Section 3.3 incorporates decision makers' preferences in the TRAP and mathematical properties of the model are discussed in Section 3.4; these are then utilized in a starting heuristic for reducing computation time; see Section 4. Section 5 presents *criterion space search* methods and the *numerical tests* performed. In Section 5.1, our suggested approach is motivated. Section 5.2 introduces a modified bi-directional ϵ-constraint method and implementation details. Section 6 presents a method to generate instances for our problem that closely represent the industrial setting, then followed by a comparison of our proposed approach with several state-of-the-art criterion space search methods on the generated instances.

2. Background and scope

The research field of production planning is broad. We provide a brief orientation of the field before diving into the specific variant of the production planning problem studied in this work. One popular way of classifying production planning models is by acknowledging the time horizons considered. This simplifies to some extent the decision variables and model parameters. Several authors (e.g., Gupta and Maranas, 1999; Min and Zhou, 2002) classify production planning problems as *strategic*, *tactical*, or *operational*. A strategic planning affects the design, the product structure, and the configuration of the factory. It is usually done two to five years in advance (the range varies among industries). Tactical planning involves determining material flows, inventory levels, and capacity utilization; it is usually done one to four years in advance and acts as an input to operational models, such as machine scheduling.

2.1. Tactical planning literature and scope of the TRAP

According to Díaz-Madroñero et al. (2014), there are five main categorizations of tactical planning problems: (a) number of products/items and structure of bill-of-materials/levels (e.g., series, assembly, general, and arborescence) (Pochet and Wolsey, 2006, Chapter 13); (b) distribution (stochastic or deterministic) of the demand; (c) time discretization; (d) capacity constraints; and (e) types of objective functions. Further, most industrial problems are multi-item and some are also multi-level. Mainly two types of time discretizations are considered: *short-time buckets*, in which their is enough time to manufacture one part/item; *long-time buckets*, in which multiple parts or an entire product is produced. A classic example in which small time buckets are used is the *discrete lot-sizing and scheduling problem* (DLSP; Lasdon and Terjung, 1971), in which the integration of the lot-sizing and scheduling problems is enhanced by the short time buckets. *Hybrid models* refer to combining large and small time buckets; a classic example of a hybrid model is the *general lot-sizing and scheduling problem* (GLSP; Fleischmann and Meyr, 1997). Transchel et al. (2011) consider a hybrid model using *macro* (discrete) and *micro* (continuous event-based) time scales. Although in most companies resources are shared by multiple products, parallel machines are considered in few articles (Wörbelauer et al., 2018; for parallel production lines). It is computationally hard to perform both scheduling and lot-sizing in a monolithic model for industrial instances. Furthermore, most companies prepare the schedule of a job only when a customer order has been created in the MRP system that is generally few hours/days (varying between companies) before the delivery time/date is fixed; so-called *rolling horizon*. Hence, long-term scheduling is generally not practically useful as modeling all the uncertainties in advance is a hard task. Therefore, there is merit in separating lot-sizing and scheduling problems. Some articles pertain to bi-objective lot sizing problems, for example, Ammar et al. (2019) (objectives inventory cost and set-up time) and Rezaei and Davoodi (2011) (objectives service level and total cost). Many articles highlight demand uncertainty, which is often modeled using stochastic programming (Lan et al., 2011; Nourelfath, 2011), less commonly by fuzzy sets (Chen and Huang, 2010; Lan et al., 2011), and by robust approaches (Wei et al., 2011; Genin et al., 2008). Most capacity constraints in tactical level models regard machines, man-hours, fixtures, tools, and inventory levels. The most common type of objective function minimizes cost/time (processing time, set-up time, and fixture costs; Bradley and Glynn, 2002; Mieghem, 2003). The drawback of using such functions is, however, that most of the cost measures rely heavily on the used accounting principles, which are generally misleading (Myrelid and Olhager, 2019).

Our model focuses on long-range resource (machine) loading, a time frame in which reliable and detailed (weekly/daily) demand predictions do not exist. Consequently, short-time buckets are not relevant. Instead our time discretization uses long-time buckets (a quarter of a year for our case) wherein it is reasonable to assume a constant material flow (i.e., precedence between jobs can be ignored) and a deterministic demand due to various fixed contracts. The manufacturing of products involves activities such as cutting, welding, heat treatment, and quality control on either the final product or parts that are later assembled into the final product. We focus only on cutting operations, which make up the majority of the total lead time of products. We propose a capacity allocation plan that promotes a balanced resource loading enabled by new qualifications. This is done by means of a bi-objective optimization model, which is to be used when one or several products are introduced in the factory or in case of significant change in available machining capacity.

2.2. *Multi-objective optimization methods*

Most real world industrial applications have several, often conflicting, objectives. This has resulted in a surge in utilization of multi-objective optimization methods to obtain so-called *efficient frontiers* (see Ehrgott, 2006). The solutions on the efficient frontier help the decision maker understand the true *trade-off* between two or more objectives. Algorithms for *bi-objective integer programming* (BOIP) and *bi-objective mixed-integer programming* (BOMIP) are broadly classified into *decision space search methods* and *criterion space search methods*. Popular methods for decision space search include evolutionary multi-objective methods (e.g., NSGA-II; Deb et al., 2002), which has gained interest although it does not provide any measure of optimality or near-optimality. There are also some nonevolutionary metaheuristics, hybrid multi-objective metaheuristics, and parallel multi-objective optimization methods that have become popular in some applications (such nonstandard approaches are discussed in Talbi et al. (2012)). Such inexact methods are, however, extremely useful if the enumeration of all the nondominated points (NDPs) using exact methods is hindered by time or memory limitations. Some improvements in branch-and-bound methods for bi-objective mixed 0–1 linear optimization problems are presented in Vincent et al. (2013).

Our work focuses on criterion space search methods, motivated by the fact that they can exploit the power of integer programming solvers (`Gurobi` in our case), and that any future computational enhancement of the solver will be replicated for the end-user of our algorithm as well. Some popular methods for criterion space search are the *weighted sum method* (Aneja and Nair, 1979), the *perpendicular search method* (Chalmet et al., 1986), the *augmented weighted Tchebycheff* (AWT) method (Bowman, 1976; Steuer and Choo, 1983), and the *ϵ-constraint method* (Miettinen, 1988, p. 85). More recently proposed methods include the *balanced box* (BB) method (Boland et al., 2015a), which extends the traditional *box method* for BOIP (Hamacher et al., 2007), and the *triangle splitting method* for BOMIP (Boland et al., 2015b). Methods such as *weighted sum* are not able to find unsupported nondominated points (Ehrgott, 2005, Definition 8.7), the existence of which makes BOMIP (and BOIP) particularly hard to solve. Most algorithms for multi-objective integer programming (MOIP) possess one basic common operation, the so-called *scalarization*, the idea of which is to transform a MOIP into a sequence of single objective IPs (integer programs). Scalarization techniques for transforming a BOIP into a sequence of IPs include the *augmentation* method, which uses appropriate values for the objective weights (Mavrotas, 2009; Özlen and Azizoğlu, 2009), and *lexicographic minimization* of the two objectives (Ehrgott, 2005, Section 5.1).

Three important factors to consider when selecting an appropriate method are (i) the maximum number of scalarized problems to solve in order to identify all the *nondominated points* (NDPs), (ii) the quality of the feasible solutions (if) available for each scalarized problem, and (iii) the computational cost of a given scalarized problem. A combination of these factors along with information about the problem type and instance sizes help in identifying a functioning approach. For example, if a problem is computationally hard to solve unless it is initialized with a good feasible solution, then a decomposition method such as the balanced box (BB; Boland et al., 2015a) and the box method (Hamacher et al., 2007) may provide better (i.e., shorter) solution times as compared to the ϵ-constraint method. However, to identify a set $\mathcal{G}$ of NDPs, the BB method solves $3|\mathcal{G}|$ scalarized problems (Boland et al., 2015a, Proposition 5), while the ϵ-constraint method solves only $2|\mathcal{G}| + 1$ scalarized problems (Chankong and Haimes, 1983). Hence, most decomposition methods solve more scalarized problems, which may result in large computing times for certain problem instances. Several attempts to combine different approaches to utilize their respective strengths and avoid weaknesses have been made. For one such example, as presented by Dai and Charkhgard (2018), the algorithm starts by employing the BB method and then switches to the ϵ-constraint method; this algorithm results in 25%–40% improvement in solution times over the pure BB method and the pure ϵ-constraint method on various popular benchmarking instances (Dai and Charkhgard, 2018, Table 1). In Leitner et al. (2016), switches between the perpendicular search and ϵ-constraint methods are done repeatedly during the course of the algorithm. The contrast between these two methods—apart from using different combinations—is that Dai and Charkhgard (2018) use a *fixed switch rule* while Leitner et al. (2016) do not. The idea of combining two or more approaches has, however, been around for a long time (to the best our knowledge, the first application appeared in Ulungu and Teghem (1994)). We conclude that different criterion space search methods have varying effects on computational performance, depending on the problem and instances.

3. Problem description and properties

The tactical resource allocation problem (TRAP) is defined as follows; notations are listed in Table 2.

Definition 1 (Tactical Resource Allocation Problem (TRAP)). *Given a set $\mathcal{J}$ of job types (tasks) and a set $\mathcal{K}$ of machines, let p_{jk} be the processing time (including set-up time) of job type $j \in \mathcal{J}$ when performed in a compatible machine $k \in \mathcal{K}_j \subseteq \mathcal{K}$. Each machine $k \in \mathcal{K}$ has the capacity C_{kt} (time units) in time period $t \in \mathcal{T}$ and a relative loading threshold $\zeta_k \in [0, 1]$. The demand a_{jt} of each job type $j \in \mathcal{J}$ in time period $t \in \mathcal{T}$ must be met. The number of machines allocated to the same job type in each time period may not exceed the value of the parameter $\tau \in \mathbb{Z}_+$. For assignments (j, k),*

Table 2
Notation for the tactical resource allocation model

Sets	Description
$\mathcal{J} = \{1, \dots, J\}$	set of job types to be performed on the products
$\mathcal{K} = \{1, \dots, K\}$	set of machines
$\mathcal{K}_j \subseteq \mathcal{K}$	set of machines feasible for job type $j \in \mathcal{J}$
$\mathcal{N}_j \subseteq \mathcal{K}_j$	set of machines feasible, but not qualified for job type $j \in \mathcal{J}$
$\mathcal{T} = \{1, \dots, T\}$	set of time-periods
Variables	**Description**
$x_{jkt} \in \mathbb{Z}_+$	number of jobs of type $j \in \mathcal{J}$ performed in machine $k \in \mathcal{K}_j$ in time period $t \in \mathcal{T}$
$s_{jkt} \in \{0, 1\}$	$= 1$ if a job of type $j \in \mathcal{J}$ is allocated to machine $k \in \mathcal{K}_j$ in time period $t \in \mathcal{T}$
$z_{jkt} \in \{0, 1\}$	$= 1$ if machine $k \in \mathcal{N}_j$ is qualified for job type $j \in \mathcal{J}$ in time period $t \in \mathcal{T}$
$n_t \in \mathbb{R}_+$	maximum resource loading above thresholds ζ_k, $k \in \mathcal{K}$, in time period $t \in \mathcal{T}$
$\mathbf{y} := (\mathbf{x}, \mathbf{s}, \mathbf{n}, \mathbf{z})$	bold notations representing vectors of the corresponding indexed variables
Parameters	**Description**
$a_{jt} \in \mathbb{Z}_+$	demand of jobs of type $j \in \mathcal{J}$ in time period $t \in \mathcal{T}$
$p_{jk} \in \mathbb{Q}_+$	machining time (including set-up time) in machine $k \in \mathcal{K}_j$ for job type $j \in \mathcal{J}$
$C_{kt} \in \mathbb{Z}_+$	capacity (hours) available in machine $k \in \mathcal{K}$ in time period $t \in \mathcal{T}$
$\beta_{jk} \in \mathbb{Z}_+$	cost associated with qualifying machine $k \in \mathcal{N}_j$ for job type $j \in \mathcal{J}$
$\gamma \in \mathbb{Z}_+$	upper limit on the number of qualifications in a single time period
$\tau \in \mathbb{Z}_+$	maximum number of alternative machines for each job type in a single time period
$\zeta_k \in [0, 1]$	loading threshold for machine $k \in \mathcal{K}$

such that $k \in \mathcal{N}_j$ and $j \in \mathcal{J}$, so-called qualifications *are required that generate additional one-time costs. It holds that $\mathcal{N}_j \subseteq \mathcal{K}_j$ for all $j \in \mathcal{J}$; for the case of a* new *job type (associated with a new product) j, $\mathcal{K}_j = \mathcal{N}_j$ holds. For a job type $j \in \mathcal{J}$, the machines in the set $\mathcal{K}_j \setminus \mathcal{N}_j$ do not require any qualifications (pre-qualified machines). The total number of qualifications performed per time period t may not exceed the value of the parameter $\gamma \in \mathbb{Z}_+$. The objectives considered are to minimize the sum (over time periods) of maximum excess resource loading above given thresholds and to minimize the qualification costs (see Section 3.1 for more details).* ■

Let us consider the example illustrated in Table 1, with two job-types, T and M. Using the notations (see Table 2) $\mathcal{J} = \{\mathrm{M}, \mathrm{T}\}$; $\mathcal{T} = \{1, 2, 3\}$; $\mathcal{N}_\mathrm{M} = \{2\}$; $\mathcal{N}_\mathrm{T} = \{3\}$; $\mathcal{K}_\mathrm{M} = \mathcal{K}_\mathrm{T} = \mathcal{K} = \{1, 2, 3\}$; $a_{\mathrm{M}1} = a_{\mathrm{M}2} = 2$; $a_{\mathrm{M}3} = 1$; $a_{\mathrm{Tk}} = 2$, $k \in \mathcal{K}$; $C_{kt} = 4$, $k \in \mathcal{K}$, $t \in \mathcal{T}$; $\zeta_k = 0.7, k \in \mathcal{K}$; $p_{\mathrm{M}1} = p_{\mathrm{M}2} = 1$; $p_{\mathrm{M}3} = 3$, $p_{\mathrm{T}1} = 3$; $p_{\mathrm{T}2} = p_{\mathrm{T}3} = 1$; $\tau = 2$; $\gamma = 1$; $\beta_{jk} = 1$, $j \in \mathcal{J}$, $k \in \mathcal{K}$. Let us denote the solution (the variable $\mathbf{x}$ in Tab. 2) illustrated in Table 1 for the job type milling (M) as $\mathbf{x}^1_{\mathrm{M}\cdot\cdot} = \left(\begin{array}{cccc} & k=1 & k=2 & k=3 \\ t=1 & 1 & 0 & 1 \\ t=2 & 1 & 1 & 0 \\ t=3 & 0 & 1 & 0 \end{array}\right)$ and for turning (T) as $\mathbf{x}^1_{\mathrm{T}\cdot\cdot} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$ (with the same row (t) and column (k) labels). The excess resource loading (loading level above the threshold ζ_k) for $t=1$ and $k=1$ is defined as $\frac{1}{C_{11}}(p_{\mathrm{M}1}x^1_{\mathrm{M}11} + p_{\mathrm{T}1}x^1_{\mathrm{T}11}) - \zeta_1 = 0.3$. The excess resource loading for each value of t (row) and k

(column) then is $\begin{pmatrix} 0.3 & 0 & 0.05 \\ 0.3 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$, which implies that for $t = 1$ the maximum resource loading over all machines is 0.3. Analogously computed, for $t = 2$ the maximum is 0.3 and for $t = 3$ the maximum is 0. Hence, the sum over the time periods of the maximum excess resource loading equals 0.6 and the qualification cost equals 2 (machine 2 is used for M and machine 3 is used for T, but $\mathcal{N}_{\mathrm{M}} = \{2\}$ and $\mathcal{N}_{\mathrm{T}} = \{3\}$). In another solution, if we do not consider qualifications, the corresponding solutions would be $\mathbf{x}^2_{\mathrm{M}\cdot\cdot} = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$ and $\mathbf{x}^2_{\mathrm{T}\cdot\cdot} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}$, which yields the excess resource loading matrix $\begin{pmatrix} 0.3 & 0 & 0.05 \\ 0.3 & 0 & 0 \\ 0.05 & 0 & 0 \end{pmatrix}$. Hence, the sum of the maximum excess resource loading equals 0.65 and the qualification cost is 0 (machine 2 is not used for M and machine 3 is not used for T).

3.1. A bi-objective mixed integer programming model of the TRAP

The constraints defining feasible resource allocations. The production should equal the demand for each job type j in each time period t, as expressed in (1a). The constraints (1b) ensure that no job is performed in any machine and time period to which it is not allocated. The constraints (1c) limit, for each job type and time period, the number of alternative allocated machines to τ, the value of which is given as input by the user; a too small value of τ may result in an empty set of feasible solutions; a too large value may lead to an increased complexity of the product routes. The constraints (1d) ensure that the machining hours do not exceed the respective machine capacities in any time period. The constraints (1e) imply that if a job of type j is assigned to machine $k \in \mathcal{N}_j$ in time period $l \in \mathcal{T}$, then machine k must be qualified for that job type *at the latest* in time period l. The constraints (1f) limit the number of scheduled qualifications in each time period to γ (due to limited number of skilled professionals for completing new qualifications). The constraints (1g)–(1j) define the allowed values of the variables x_{jkt}, s_{jkt}, z_{jkt}, and n_t. The feasible solutions to the TRAP are thus defined as the variable vectors $(\mathbf{x}, \mathbf{s}, \mathbf{n}, \mathbf{z})$ fulfilling the constraints

$$\sum_{k \in \mathcal{K}_j} x_{jkt} = a_{jt}, \qquad j \in \mathcal{J},\ t \in \mathcal{T}, \tag{1a}$$

$$x_{jkt} \le \min\left\{a_{jt}, \left\lfloor \frac{C_{kt}}{p_{jk}} \right\rfloor\right\} s_{jkt}, \qquad k \in \mathcal{K}_j,\ j \in \mathcal{J},\ t \in \mathcal{T}, \tag{1b}$$

$$\sum_{k \in \mathcal{K}_j} s_{jkt} \le \tau, \qquad j \in \mathcal{J},\ t \in \mathcal{T}, \tag{1c}$$

$$\frac{1}{C_{kt}} \sum_{j \in \mathcal{J}} p_{jk} x_{jkt} \le n_t + \zeta_k \le 1, \qquad k \in \mathcal{K},\ t \in \mathcal{T}, \tag{1d}$$

$$\sum_{t \in \mathcal{T}: t \le l} z_{jkt} \ge s_{jkl}, \qquad k \in \mathcal{N}_j,\ j \in \mathcal{J},\ l \in \mathcal{T}, \tag{1e}$$

$$\sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{N}_j} z_{jkt} \le \gamma, \qquad t \in \mathcal{T}, \tag{1f}$$

$$x_{jkt} \in \mathbb{Z}_+, \qquad k \in \mathcal{K}_j,\ j \in \mathcal{J},\ t \in \mathcal{T}, \tag{1g}$$

$$s_{jkt} \in \{0, 1\}, \qquad k \in \mathcal{K}_j,\ j \in \mathcal{J},\ t \in \mathcal{T}, \tag{1h}$$

$$z_{jkt} \in \{0, 1\}, \qquad k \in \mathcal{N}_j,\ j \in \mathcal{J},\ t \in \mathcal{T}, \tag{1i}$$

$$n_t \geq 0, \qquad t \in \mathcal{T}. \tag{1j}$$

Definition 2 (Set of feasible solutions and its image in the criterion space). *Defining*[2] $\mathbf{y} := (\mathbf{x}, \mathbf{s}, \mathbf{n}, \mathbf{z})$, *for any values of* $\tau, \gamma \in \mathbb{Z}_+$, *the set of feasible solutions to the model* (1) *is denoted as*

$$Y(\tau, \gamma) := \big\{ \mathbf{y} \,\big|\, \textit{the constraints } (1a)\text{–}(1j) \textit{ hold} \big\}. \qquad \blacksquare$$

The two objectives considered represent the preferences of the planners. The first objective is to *minimize the excess resource loading*, expressed as to

$$\underset{\mathbf{x},\mathbf{s},\mathbf{n},\mathbf{z}}{\text{minimize}} \qquad g_1(\mathbf{x}, \mathbf{s}, \mathbf{n}, \mathbf{z}) := \sum_{t \in \mathcal{T}} n_t, \tag{2a}$$

where n_t is limited by the constraints (1d) and (1j).

It considers the sum over the time periods $t \in \mathcal{T}$ of the *excess resource loading of the machines* (i.e., $n_t \geq 0$), which is defined as the maximum (over the machines) ratio between the allocated machining hours and the available hours (i.e., $\frac{1}{C_{kt}} \sum_{j \in \mathcal{J}} p_{jk} x_{jkt}$) minus the loading threshold $\zeta_k \in [0, 1]$ for the machine. Therefore, in a solution $(\mathbf{x}, \mathbf{s}, \mathbf{n}, \mathbf{z})$ that minimizes the objective g_1, the equality $n_t = \max\{0; \max_{k \in \mathcal{K}}\{\frac{1}{C_{kt}} \sum_{j \in \mathcal{J}} p_{jk} x_{jkt} - \zeta_k\}\}$ will hold for every $t \in \mathcal{T}$. In the context of a bi-objective mixed integer programming (BOMIP), this objective is thus defined by (2a), (1d), (1g), and (1j). The constraints (1d) prevent the loading level of each machine in each time period from exceeding one, hence, imposing a capacity limit on each machine. Minimizing the excess resource loading will result in an increased capacity of buffers, which in turn enables reduced lead times for the products.

An alternative to the min-sum objective in (2a), which amounts to solving $\min_{\mathbf{y} \in Y(\tau,\gamma)}\{\sum_{t \in \mathcal{T}} n_t\}$, would be to choose a min–max objective, that is, to solve $\min_{\mathbf{y} \in Y(\tau,\gamma)}\{\max_{t \in \mathcal{T}} n_t\}$. The latter may result in more reasonable solutions, but is not chosen due to two reasons. (i) First, since there is no clear priority among the excess resource loading n_t over the time periods $t \in \mathcal{T}$, let us hypothetically consider T objective functions (n_t) defining a T-dimensional criterion space. While the min-sum objective is guaranteed to always yield an efficient solution in this T-criterion space, the min–max objective (also known as *max-ordering*) is *not* (Ehrgott, 2005, Proposition 9). The min–max objective can, however, be utilized to calculate an efficient solution by performing a *lexicographic max-ordering optimization* (Ehrgott, 2005, Chapter 5.3). But since the order in which the objectives are minimized is not known *a priori* (generally $T \geq 16$ holds for all our problem instances) identifying a nonincreasing ordered sequence of the objective functions' values will significantly increase the computing time. (ii) From a practical standpoint, in some time periods the

[2]The notations $(\mathbf{x}, \mathbf{s}, \mathbf{n}, \mathbf{z})$ and $\mathbf{y}$ will be used interchangeably throughout this article.

demand for products might be consistently higher, leading to higher loading levels of the machining resources. Hence, a min–max objective—focusing on high-demand time periods and ignoring the others—can be counterintuitive for production planners.

The second objective is to *minimize the total qualification cost*, defined as the sum of the one-time costs incurred by qualifying machines for job types, over the time periods. It is expressed as to

$$\underset{\mathbf{x},\mathbf{s},\mathbf{n},\mathbf{z}}{\text{minimize}} \qquad g_2(\mathbf{x},\mathbf{s},\mathbf{n},\mathbf{z}) := \sum_{t\in\mathcal{T}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{N}_j} \beta_{jk} z_{jkt}. \tag{2b}$$

Note that an increase in the number of qualifications (g_2) will enable a reduction of the excess loading of the machines (g_1).

The vector-valued function of objectives is denoted as $\mathbf{g}(\mathbf{y}) := (g_1(\mathbf{y}), g_2(\mathbf{y}))$, and the image of the set $Y(\tau,\gamma)$ in the criterion space is defined as $\mathbf{g}(Y(\tau,\gamma)) := \{\,\mathbf{g}(\mathbf{y}) \in \mathbb{R}^2 \mid \mathbf{y} \in Y(\tau,\gamma)\,\}$.

Definition 3 (Efficient solution). *A solution* $\bar{\mathbf{y}} \in Y(\tau,\gamma)$ *is an* efficient solution *to the TRAP if* $\nexists$ $\mathbf{y} \in Y(\tau,\gamma)$ *such that the relations* $\mathbf{g}(\mathbf{y}) \leq \mathbf{g}(\bar{\mathbf{y}})$ *and* $\mathbf{g}(\mathbf{y}) \neq \mathbf{g}(\bar{\mathbf{y}})$ *hold. If* $\bar{\mathbf{y}}$ *is efficient in the decision space, then* $\mathbf{g}(\bar{\mathbf{y}})$ *is a* nondominated point *in the criterion space. The set of efficient solutions to the bi-objective optimization problem (1)–(2) is denoted as* $Y_{\text{eff}}(\tau,\gamma)$. ■

3.2. Computational complexity

We show that the TRAP (Definition 1) is NP-hard, by comparing with the *makespan minimization problem*.

Definition 4 (Makespan minimization). *Given n jobs and m machines, let* d_{jk} *be the processing time of job* $j \in \{1,\dots,n\}$ *if it is assigned to machine* $k \in \{1,\dots,m\}$. *No machine can process two jobs at the same time. After a job has started getting processed, it must be completed on that machine without any interruption. The makespan minimization of unrelated parallel machine is the problem of assigning each job to a machine such that the maximum completion time (makespan)* C_{max} *over the machines is minimized. This problem is referred to as* $R||C_{max}$*; see (Lawler et al., 1993, Chapter 9). The problem is NP-hard in the strong sense, since its version where* $d_{jk} = d$*, that is, the problem denoted* $P||C_{max}$*, is also NP-hard; Garey and Johnson (1979).* ■

Proposition 1 (Problem reduction). *The makespan minimization of the unrelated parallel machine scheduling problem, that is,* $R||C_{max}$*, is polynomially reducible to the TRAP of Definition 1.*

Proof. Letting, for $j \in \{1,\dots,n\}$ and $k \in \{1,\dots,m\}$, $\ell_{jk} = 1$ if job j is assigned to machine k, and $\ell_{jk} = 0$ otherwise, the problem $R||C_{\max}$ is modeled as

$$\min_{\ell}\left\{ \max_{k\in\{1,\dots,m\}}\left\{\sum_{j=1}^{n} d_{jk}\ell_{jk}\right\} \;\middle|\; \sum_{k=1}^{m}\ell_{jk} = 1;\ \ell_{jk}\in\{0,1\}, k\in\{1,\dots,m\}, j\in\{1,\dots,n\}\right\}. \tag{3}$$

Consider then an instance of TRAP (see Definition 1) given by the following: $T = 1$; $J = n$; $K = \tau = m$; $\mathcal{K}_j = \mathcal{K}$, $\mathcal{N}_j = \emptyset$, $a_{j1} = 1$, $j \in \mathcal{J}$; $\sum_{j=1}^{n} p_{jk} \leq C_{k1} = C > 0$, $\zeta_k = 0$, $k = 1,\dots,m$.

The corresponding special case of our model (1)–(2), is then expressed as the single-objective MIP to

$$\underset{\mathbf{x}, n_1}{\text{minimize}} \quad n_1, \tag{4a}$$

$$\text{subject to} \quad \sum_{k=1}^{m} x_{jk1} = 1, \qquad j = 1, \dots, n, \tag{4b}$$

$$\sum_{j=1}^{n} \frac{p_{jk}}{C} x_{jk1} \leq n_1, \qquad k = 1, \dots, m, \tag{4c}$$

$$x_{jk1} \in \{0, 1\}, \qquad j = 1, \dots, n, \; k = 1, \dots, m. \tag{4d}$$

The instance of TRAP, expressed in (4), is an $R||C_{\max}$, as expressed in (3). An optimal solution $(\mathbf{x}^*, n_1^*)$ to (4) is equivalent to an optimal solution to this instance of the TRAP (1)–(2) and given by $(\mathbf{x}, \mathbf{s}, n_1, \mathbf{z}) = (\mathbf{x}^*, \mathbf{1}, n_1^*, -) \in Y(m, \gamma)$. Since the $R||C_{\max}$ decision problem is NP-complete, it follows that the TRAP is NP-hard. ■

3.3. Considering a third objective: routing efficient solutions

The constraints (1c) limit the number of alternative machines allocated to a job type in each time period to τ, the practical effect of which is reducing the number of possible routings (i.e., sequences in which machines are visited) used by a product. A decision maker prefers efficient solutions to the bi-objective model (1)–(2) that make use of low numbers of alternative machines. Hence, we define the set of *routing efficient solutions*, $Y_{\text{reff}}(\tau, \gamma) \subseteq Y_{\text{eff}}(\tau, \gamma)$ (see Definition 3), by means of the function $g_3(\mathbf{y}) := \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}_j} s_{jkt}$, representing the total number of alternative machines used. Assigning g_3 a lower priority than g_1 and g_2, its inclusion will not change the nondominated points (NDPs) in the two-dimensional criterion space to the TRAP. The less importance of g_3 as compared to g_1 and g_2 is because g_3 is associated with internal transportation cost that has less priority.

Definition 5 (Routing efficient solution). *A solution* $\bar{\mathbf{y}} \in Y_{\text{eff}}(\tau, \gamma)$ *is* routing efficient*, that is,* $\bar{\mathbf{y}} \in Y_{\text{reff}}(\tau, \gamma) \subseteq Y_{\text{eff}}(\tau, \gamma)$*, if and only if* $\nexists \mathbf{y} \in Y(\tau, \gamma)$ *such that the relations* $g_i(\mathbf{y}) = g_i(\bar{\mathbf{y}}), i \in \{1, 2\}$ *, and* $g_3(\mathbf{y}) < g_3(\bar{\mathbf{y}})$ *hold.* ■

For the BOMIP (1)–(2) with parameter values τ and γ, all the *nondominated points* (NDPs) in the objective (i.e., criterion) space (see Section 2.2) lie in the rectangle

$$B(\mathbf{g}^{\text{TOP}}, \mathbf{g}^{\text{BOT}}) := \left\{ \mathbf{g} \in \mathbb{R}^2 \,\middle|\, g_1^{\text{TOP}} \leq g_1 \leq g_1^{\text{BOT}}, \; g_2^{\text{BOT}} \leq g_2 \leq g_2^{\text{TOP}} \right\},$$

where $(g_1^{\text{TOP}}, g_2^{\text{BOT}})^\top$ and $(g_1^{\text{BOT}}, g_2^{\text{TOP}})^\top$ denote the *ideal* and *nadir* points, respectively (Miettinen, 1988, p. 15–16), and the area of which is $A(\mathbf{g}^{\text{TOP}}, \mathbf{g}^{\text{BOT}}) := (g_1^{\text{BOT}} - g_1^{\text{TOP}})(g_2^{\text{TOP}} - g_2^{\text{BOT}})$. The solutions corresponding to the points $\mathbf{g}^{\text{TOP}}$ and $\mathbf{g}^{\text{BOT}}$ are computed by lexicographic minimization (Ehrgott, 2005, Section 5.1), as

$$\mathbf{y}^{\mathrm{TOP}} := \underset{\mathbf{y}\in Y(\tau,\gamma)}{\arg\operatorname{lexmin}} \{g_1(\mathbf{y}) + w_2 g_2(\mathbf{y}),\ g_3(\mathbf{y})\}, \tag{5a}$$

$$\mathbf{y}^{\mathrm{BOT}} := \underset{\mathbf{y}\in Y(\tau,\gamma)}{\arg\operatorname{lexmin}} \{g_2(\mathbf{y}) + w_1 g_1(\mathbf{y}),\ g_3(\mathbf{y})\}, \tag{5b}$$

where the parameters $w_1, w_2 > 0$ are set appropriately (Özlen and Azizoğlu, 2009, Theorem 2.2) to yield (at least weakly) efficient solutions with respect to the objectives in (2).

3.4. Integrality of the variables **z**

For fixed values of the binary variables **s**, we define the following polyhedron in the **z**-variable space:

$$Z(\mathbf{s},\gamma) := \left\{ \mathbf{z} \,\middle|\, z_{jkt} \in [0,1],\ k \in \mathcal{N}_j,\ j \in \mathcal{J},\ t \in \mathcal{T},\ \text{and (1e)–(1f) hold} \right\}. \tag{6}$$

Proposition 2 (Integrality of the variables **z**). *For any $s_{jkt} \in \{0,1\}$, $k \in \mathcal{N}_j$, $j \in \mathcal{J}$, $t \in \mathcal{T}$, all extreme points to the polyhedron $Z(\mathbf{s},\gamma)$, defined in (6), are integral.*

Proof. The set $Z(\mathbf{s},\gamma)$ is defined by the constraints (1e), (1f), and

$$z_{jkt} \in [0,1], \qquad k \in \mathcal{N}_j,\ j \in \mathcal{J},\ t \in \mathcal{T}. \tag{7}$$

Consider any given $s_{jkt} \in \{0,1\}$, $j \in \mathcal{J}$, $k \in \mathcal{N}_j$, $t \in \mathcal{T}$. Then, for any given pair (j,k), where $k \in \mathcal{N}_j$ and $j \in \mathcal{J}$, define $T_{jk}(\mathbf{s}) := \min\{ t \in \mathcal{T} \mid s_{jkt} = 1 \}$; it follows that, among the constraints in (1e), only those corresponding to the index triples $(j, k, T_{jk}(\mathbf{s}))$ (i.e., the tightest ones) are necessary for defining the set $Z(\mathbf{s},\gamma)$ (for the case when $s_{jk} = 0$, $t \in \mathcal{T}$, all the corresponding constraints in (1e) can be removed). Hence, the polyhedron $Z(\mathbf{s},\gamma)$ can be expressed by the constraints (1f), (7), and

$$\sum_{l=1}^{T_{jk}(\mathbf{s})} z_{jkl} \geq 1, \qquad k \in \mathcal{N}_j,\ j \in \mathcal{J}. \tag{8}$$

The constraint matrix defined by (1f), (8) admits an equitable row bicoloring, hence it is totally unimodular, which implies that the polyhedron $Z(\mathbf{s},\gamma)$ has integral extreme points (Conforti et al., 2014, Corollary 4.8, Theorem 4.5, p. 133–134). The proposition follows. ■

3.5. Property of the efficient frontier for the TRAP model

The efficient frontier of a general BOMIP has isolated points as well as closed, half-open, and open line segments; (Boland et al., 2015a, Theorem 3). For the TRAP model, however, the following proposition holds.

Proposition 3 (Efficient frontier for the TRAP model)**.** *The efficient frontier of the model (1)–(2) contains only isolated nondominated points (NDPs), and no (closed, half open, or open) line segments, irrespective of the values of the parameters β_{jk}, $k \in \mathcal{N}_j$, $j \in \mathcal{J}$.*

Proof. It is sufficient to show that no line segment between any two NDPs in the criterion space exists. First, in any efficient solution, the variables $(\mathbf{x}, \mathbf{s}, \mathbf{z})$ will take finite integer values, say $(\mathbf{x}', \mathbf{s}', \mathbf{z}')$. Hence, from (1j) and (1d) follow that each variable n_t will either take the value $n'_t = 0$ or $n'_t = \max_{k \in \mathcal{K}}\{\frac{1}{C_{kt}} \sum_{j \in \mathcal{J}} p_{jk} x'_{jkt} - \zeta_k\}$, that is, n'_t takes only discrete values. Hence, the efficient solutions constitute a discrete set, which maps to a discrete set of (finite) NDPs in the criterion space. The proposition follows. ■

4. The starting heuristic

For solving the BOMIP (1)–(2) , we use a MILP solver, the efficiency of which partly relies on an early pruning of branches owing to high-quality initial feasible solutions. We propose a simple heuristic that either quickly finds a feasible solution, returns no solution (but does not conclude infeasibility), or concludes that the problem is infeasible. A feasible solution (if found) is provided to the MILP solver while looking for a nondominated point $\mathbf{g}^{\mathrm{TOP}}$ (see Section 3.3). The heuristic relies on a decomposition of the model (1)–(2) with respect to the time periods resulting in one (smaller) MILP per time period.[3] Since we aim at a solution with an objective vector value as close as possible to $\mathbf{g}^{\mathrm{TOP}}$, the heuristic employs an objective function which prioritizes minimizing the objective function g_1, defined in (2a). The heuristic is described below and summarized in Algorithm 1. Let the vector $\mathbf{y}^t := (\mathbf{x}^t, \mathbf{s}^t, n^t, \mathbf{z}^t)$ represent the corresponding variables for $t \in \mathcal{T}$. The constraints (1e) connect the time periods—thus complicating the model—and are replaced by the time-disconnected constraints

$$z^t_{jk} \geq \xi^{t-1}_{jk} s^t_{jk}, \qquad k \in \mathcal{N}_j,\ j \in \mathcal{J},\ t \in \mathcal{T}, \tag{9}$$

where the constants $\xi^{t-1}_{jk} \in \{0, 1\}$ are computed according to (12). Sequentially for $t = 1, \ldots, T$, the time-disconnected feasible sets are then defined as[4]

$$Y^t(\xi^{t-1}, \gamma) := \left\{ \mathbf{y}^t \,\middle|\, (1\text{a})^t\text{–}(1\text{d})^t,\ (9)^t,\ (1\text{f})^t\text{–}(1\text{j})^t \right\}. \tag{10}$$

[3]Another approach to finding a good starting feasible solution would be to use a Lagrangian heuristic, where, for example, the constraints (1e) are relaxed, resulting in the subproblem separating over the time indices $t \in \mathcal{T}$ as well as between the variables $(\mathbf{x}, \mathbf{s}, \mathbf{n})$ and $\mathbf{z}$. Each of the $2T$ subproblems would still be NP-hard, but considerably smaller than the TRAP. A Lagrangian heuristic does, however, require several dual iterations to find good enough dual variable values and, finally, a feasibility heuristic.

[4]For brevity, we define the notation (constraint reference)^t, referring solely to the time index t of the corresponding constraints.

Algorithm 1. Starting heuristic

```
Data: values of the parameters in Table 2
Result: a feasible solution, no solution, or an infeasibility conclusion
$t := 1, \xi^0_{jk} := 1, k \in \mathcal{N}_j, j \in \mathcal{J}$ ;
while $t \leq T$ do
    Solve the model $(11)^t \Longrightarrow \bar{\mathbf{y}}^t = (\bar{\mathbf{x}}^t, \bar{\mathbf{s}}^t, \bar{n}^t, \bar{\mathbf{z}}^t)$ ;
    if $\bar{\mathbf{y}}^t$ feasible then
        restore $\gamma$ ; compute $\xi^t_{jk}$ from (12) ; $t := t + 1$
    else
        if $t > 1$ then
            if $(1\mathrm{f})^t$ is in an irreducible inconsistent subsystem then
                $\gamma := \gamma + \Delta$, where $\Delta \geq 1$
            else
                return "model (1) is infeasible"; break
            end
        else
            return "model (1) is infeasible"; break
        end
    end
end
Solve the model (13) $\Longrightarrow \hat{\mathbf{z}} := (\hat{\mathbf{z}}^t)_{t\in\mathcal{T}}$ ;
if $\hat{\mathbf{z}}$ is feasible in (13) then
    return "$(\bar{\mathbf{x}}, \bar{\mathbf{s}}, \bar{\mathbf{n}}, \hat{\mathbf{z}})$ is feasible in model (1)"
else
    return "no feasible solution found"
end
```

Define $g^t_1(\mathbf{y}^t) := n^t$, $g^t_2(\mathbf{y}^t) := \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{N}_j} \beta_{jk} z^t_{jk}$, and $g^t_3(\mathbf{y}^t) := \sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}_j} s^t_{jk}$. The optimal set to the time-disconnected problem at time step t is then defined as

$$\bar{Y}^t(\xi^{t-1}, \gamma) := \underset{\mathbf{y}^t \in Y^t(\xi^{t-1},\gamma)}{\arg\operatorname{lexmin}} \left\{ g^t_1(\mathbf{y}^t) + w_2 g^t_2(\mathbf{y}^t),\ g^t_3(\mathbf{y}^t) \right\}, \qquad t = 1, \dots, T, \tag{11}$$

where the *weight parameter* $w_2 > 0$ is small enough such that the resulting solutions are (at least weakly) efficient (Özlen and Azizoğlu, 2009, Theorem 2.2) with respect to the two objectives $g^t_1(\mathbf{y}^t)$ and $g^t_2(\mathbf{y}^t)$; the operator lexmin is defined as in Section 5.1 in Ehrgott (2005). The output of (11) is a routing efficient solution (see Section 3.3 and Definition 5) for time period t with respect to the feasible set $Y^t(\xi^{t-1}, \gamma)$.

Sequentially, for $t = 1, \dots, T$, given the values of $\bar{z}^t_{jk}$ in $(\bar{\mathbf{x}}^t, \bar{\mathbf{s}}^t, \bar{n}^t, \bar{\mathbf{z}}^t) = \bar{\mathbf{y}}^t \in \bar{Y}^t(\xi^{t-1}, \gamma)$, the coefficients ξ^t_{jk}, $k \in \mathcal{N}_j$, $j \in \mathcal{J}$, are computed as $\xi^0_{jk} := 1$ and

$$\xi^t_{jk} := \begin{cases} 0, & \text{if } \left(\bar{z}^t_{jk} = 1 \ \wedge\ \xi^{t-1}_{jk} = 1\right) \vee \xi^{t-1}_{jk} = 0, \\ 1, & \text{otherwise,} \end{cases} \qquad t = 1, \dots, T-1. \tag{12}$$

If $Y^1(\xi^0, \gamma) = \emptyset$, then there is no feasible solution to the model (1)–(2), that is, $Y(\tau, \gamma) = \emptyset$. This is due to the qualification capacity γ being too low for the qualifications necessary in time period 1.

If, for $t = 1$, a feasible solution to the model $(11)^t$ is found, the index t is increased by one and the model $(11)^t$ is solved with ξ^{t-1} as input and $\bar{\mathbf{y}}^t$ as output. A solution obtained for a certain value of t may not be changed; hence, for any $t > 1$, the model $(11)^t$ may be infeasible even if the original feasible set—as defined in (1)—is nonempty. For an infeasible model $(11)^t$, feasibility is achieved by an increase of the value of γ (temporarily, for this value of t). The feasibility loop terminates when either a feasible solution to the model $(11)^t$ is found or the constraint $(1\text{f})^t$ is *not* in an *irreducible inconsistent subsystem*[5] (IIS); see Parker and Ryan (1996, Section 2). In the latter case, the original model (1) is infeasible, since neither the constraints $(1\text{e})^t$ nor $(1\text{f})^t$ can be in IIS, which implies that one of the other time-disconnected constraints $(1\text{a})^t$–$(1\text{d})^t$ or variable bounds $(1\text{g})^t$–$(1\text{j})^t$ are in IIS, which cannot be resolved. Hence, the model is infeasible. After exiting the outer loop the values $[\bar{\mathbf{s}}^t]_{t\in\mathcal{T}}$ are used for resolving any infeasibility in $[\bar{\mathbf{z}}^t]_{t\in\mathcal{T}}$ through the *time-connected* linear program (LP) (see Proposition 2) defined as

$$\hat{Z}([\bar{\mathbf{s}}^t]_{t\in\mathcal{T}}, \gamma) := \underset{z\in Z([\bar{\mathbf{s}}^t]_{t\in\mathcal{T}}, \gamma)}{\arg\min} \; \{g_2(\mathbf{y})\}, \tag{13}$$

where the set $Z([\bar{\mathbf{s}}^t]_{t\in\mathcal{T}}, \gamma)$ is defined analogously as in (6). The LP (13) can be solved efficiently by any commercial solver; if infeasible, however, the heuristic terminates without any conclusion about the feasibility of the model (1).

5. Criterion space search method

According to Proposition 3, the TRAP has only isolated NDPs. With some care for the respective algorithmic properties, we may now use ideas from BOIP algorithms to find (almost) all NDPs of the TRAP (1)–(2).

5.1. Search for nondominated points by the ϵ-constraint and AWT methods: strengths and weaknesses

We briefly describe the ϵ-constraint and the AWT methods along with their main properties; below, h_1 and h_2 denote the objective functions, $\mathbf{h}(\mathbf{x}) = (h_1(\mathbf{x}), h_2(\mathbf{x}))^\top$, $\mathbf{h}^{\text{ref}}$ is a reference point, $\mathbf{1} = (1, 1)^\top$, and X is an integer set.

The ϵ-constraint method yields the scalarization for a BOIP as $\min_{\mathbf{x}\in X}\{h_1(\mathbf{x}) \mid h_2(\mathbf{x}) \le \epsilon\}$ or $\min_{\mathbf{x}\in X}\{h_2(\mathbf{x}) \mid h_1(\mathbf{x}) \le \epsilon\}$. An optimal solution to a given scalarization is guaranteed to be weakly efficient; Chankong and Haimes (1983). Further, all efficient solutions can be found by setting appropriate values for ϵ. Since no feasible solution is available when solving a given scalarized problem, computing times may be longer as compared to the case of an available feasible solution. This issue can be resolved by a bi-directional (BD) variant of the ϵ-constraint method, denoted the BD-ϵ method; see (Boland et al., 2015a, Section 5.1) and Algorithm 2. Then, the ϵ-constraint method is applied alternately to the two objectives, such that a previously identified efficient solution is

[5]An IIS is a subset of the constraints of a model such that (a) the model including all constraints in the subset is infeasible, and (b) the model including all but one constraint in the subset is feasible.

Algorithm 2. Modified bi-directional ϵ-constraint

```
Data: $\tau, \gamma \in \mathbb{Z}_+$; $Y(\tau,\gamma)$; $\mathbf{g}^{\text{TOP}}$; $\mathbf{g}^{\text{BOT}}$; $\phi \in (0,1)$; $\psi_1 > 0$; $\psi_2 = 1$; $w_1, w_2 > 0$; $t_{\text{lim}}$
Result: NDP                                             // set of non-dominated points
$\mathbf{g}^1 := \mathbf{g}^{\text{TOP}}$ ; $\mathbf{g}^2 := \mathbf{g}^{\text{BOT}}$ ; NDP := $\{\mathbf{g}^1, \mathbf{g}^2\}$ ; area := $A(\mathbf{g}^1, \mathbf{g}^2)$ ;
while ( $(g_1^2 - g_1^1 > \psi_1) \wedge (g_2^1 - g_2^2 > \psi_2)$ ) do
    if ( area $\leq \phi \cdot A(\mathbf{g}^{\text{TOP}}, \mathbf{g}^{\text{BOT}})$ ) then
        NDP := NDP $\cup$ AWT($\mathbf{g}^1, \mathbf{g}^2$) ;
        break ;
    end
    $\mathbf{g}^1 := \operatorname{lexmin}_{\mathbf{y} \in Y(\tau,\gamma)} \{g_1(\mathbf{y}) + w_2 g_2(\mathbf{y}), g_3(\mathbf{y}) : g_2(\mathbf{y}) \leq g_2^1 - \psi_2\}$      // model TOP
    if ( (optimality verified in $t_{\text{lim}}$) $\wedge$ ($\mathbf{g}^1 \neq \mathbf{g}^2$) ) then
        NDP := NDP $\cup$ $\mathbf{g}^1$ ; area := $A(\mathbf{g}^1, \mathbf{g}^2)$ ;
    else
        break ;
    end
    $\mathbf{g}^2 := \operatorname{lexmin}_{\mathbf{y} \in Y(\tau,\gamma)} \{g_2(\mathbf{y}) + w_1 g_1(\mathbf{y}), g_3(\mathbf{y}) : g_1(\mathbf{y}) \leq g_1^2 - \psi_1\}$      // model BOT
    if ( (optimality verified in $t_{\text{lim}}$) $\wedge$ ($\mathbf{g}^2 \neq \mathbf{g}^1$) ) then
        NDP := NDP $\cup$ $\mathbf{g}^2$ ; area := $A(\mathbf{g}^1, \mathbf{g}^2)$ ;
    else
        break ;
    end
end
```

available when solving the scalarized problem. The bi-directional approach does not require solving an infeasible problem as the last scalarized problem, cf. the original ϵ-constraint method.

The AWT method yields the scalarization for a BOIP as $\min_{\mathbf{x} \in X}\{\max\{\alpha_1(h_1(\mathbf{x}) - h_1^{\text{ref}}); \alpha_2(h_2(\mathbf{x}) - h_2^{\text{ref}})\} + \lambda \mathbf{1}^\top(\mathbf{h}(\mathbf{x}) - \mathbf{h}^{\text{ref}})\}$. The method recursively searches for a yet-unknown nondominated point by minimizing the maximum weighted (by $\boldsymbol{\alpha} \in \mathbb{R}_+^2$) distance (the l_∞-norm) from $\mathbf{h}^{\text{ref}}$. The second term (the l_1-norm) is added to avoid generating too many weakly nondominated points (NDPs), using a suitable value of $\lambda \geq 0$. The AWT method can identify all the NDPs and is used in many interactive multi-objective optimization approaches (Steuer and Choo, 1983). Dächert et al. (2012) establish that too small values of λ cause numerical difficulties, while too large values may result in oversight of some NDPs; they also derive problem dependent (adaptive) formulae for calculating α_1, α_2, and λ; these are, however, applicable only for pure BOIPs with integer valued objective functions. Dächert et al. (2012) has shown computational superiority of their adaptive formulae (for the algorithm presented in Ralphs et al. (2006)) over using fixed parameters instead. Steuer and Choo (1983) present a *lexicographic weighted Tchebycheff method* that guarantees that all NDPs are found, without requiring input parameters α_1, α_2, and λ; it requires, however, the solution of two optimization problems and is computationally expensive, as empirically verified by Miettinen et al. (2006); to linearize the max-function additional variables and constraints are included, which may increase the solution times as well.

As discussed in Ehrgott (2006, Section 4.4), a drawback of the ϵ-constraint method for solving a BOIP is that the ϵ-constraint is a knapsack constraint; this may reduce the computational tractability of the scalarized problem (Ehrgott, 2006, Sections 2.2 & 3.2). In a branch-and-bound tree corresponding to a given ϵ-constraint scalarization, the contradictory nature of the two objectives implies that the ϵ-constraint will be active in an optimal solution at several nodes if possible; hence, many branch-and-bound nodes need be explored before reaching a node with inactive ϵ-constraints (see Appendix A.1).

Ehrgott (2006, Section 4) suggests replacing the ϵ-constraint by a so-called *elastic constraint*. The idea is to allow violation of the ϵ-constraint at the expense of a weighted penalty variable added to the objective function. The approach has been successful in several applications (see, e.g., Ehrgott and Ryan, 2002 for an implementation for a *crew-scheduling problem*). Finding appropriate values for the coefficient of the penalty variable is, however, tricky and the computational efficiency depends hugely on the type of problem and instance.

We propose and investigate a modified version of the bi-directional ϵ-constraint method, in which the second stage uses the reference-point based AWT method, which also avoids the ϵ-constraints (an example showing one of the differences between the scalarization of the two methods is highlighted in Appendix A.1).

5.2. Implementation of the modified bi-directional ϵ-constraint method

Our proposed approach, the *modified bi-directional ϵ-constraint method*, combines the BD-ϵ method with the AWT method; it is described below and summarized in Algorithm 2.

Description of the algorithm. The input to Algorithm 2 are the two initial nondominated points $\mathbf{g}^{\text{TOP}}$ and $\mathbf{g}^{\text{BOT}}$, defined in (5), and which constitute the initial points in the set `NDP` of nondominated points. The two points $\mathbf{g}^{\text{TOP}}$ and $\mathbf{g}^{\text{BOT}}$ are used to initialize $\mathbf{g}^1$ and $\mathbf{g}^2$, respectively. The parameter $\phi \in (0, 1)$ determines when to switch from the BD-ϵ method to the AWT, while the parameter ψ_1 (ψ_2) denotes the value by which g_1^2 (g_2^1) is reduced while applying the ϵ-constraint on $g_1(\mathbf{y})$ ($g_2(\mathbf{y})$).

Since the qualification cost parameter β_{jk} is integral, we set $\psi_2 := 1$. We propose switching to the AWT method if the area of the rectangle $B(\mathbf{g}^1, \mathbf{g}^2)$ to be explored for yet-unknown nondominated points fulfills $A(\mathbf{g}^1, \mathbf{g}^2) \leq \phi \cdot A(\mathbf{g}^{\text{TOP}}, \mathbf{g}^{\text{BOT}})$ (see Fig. 1). We made computational tests for $\phi \in \{0.25, 0.35\}$ and $\psi_2 = 0.01$, which are reasonable for our problem instances. We set a time limit, t_{lim} of 5000 seconds for solving each scalarized problem, after which the solver terminates the computations (see Section 6, §1 for details about the termination criteria used).

Algorithm 2 starts by the BD-ϵ method (i.e., if $A(\mathbf{g}^1, \mathbf{g}^2) > \phi \cdot A(\mathbf{g}^{\text{TOP}}, \mathbf{g}^{\text{BOT}})$), which is expressed by the scalarizations referred to as TOP and BOT; TOP and BOT apply an ϵ-constraint on the qualification cost and the excess resource loading, respectively. The model TOP is solved and the value of $\mathbf{g}^1$ is updated; if the solver does not verify optimality in a user-defined time limit it terminates (see Sections 6 and 1 for criteria to verify optimality). The model BOT is solved and the value of $\mathbf{g}^2$ is updated. (In Fig. 1, $g_1^2 = g_1^{\text{BOT}} = 2.48$ and $\psi_1 = 0.01$, while $g_2^1 = g_2^{\text{TOP}} = 24$ and $\psi_2 = 1$.) If the area of the remaining rectangle fulfills $A(\mathbf{g}^1, \mathbf{g}^2) \leq \phi \cdot A(\mathbf{g}^{\text{TOP}}, \mathbf{g}^{\text{BOT}})$, Algorithm 2 switches to the AWT method, which is provided two nondominated points, $\mathbf{g}^1$ and $\mathbf{g}^2$. The AWT method searches for yet-unknown nondominated points in the interior of $B(\mathbf{g}^1, \mathbf{g}^2)$; cf. Section 5.1 The adaptive formulae used to calculate $\boldsymbol{\alpha}$ and λ are found in (Dächert et al., 2012, Table 2), and are guaranteed to yield only efficient solutions when both objective functions take only integer values (for this purpose, one

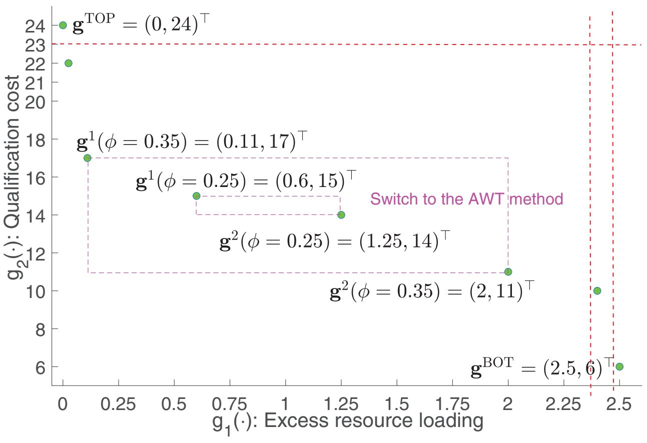

Fig. 1. BD-ϵ with switch to the AWT. The green points are nondominated, the red-dashed lines represent ϵ-constraints, and the purple-dashed rectangles represent the switch to the AWT method for $\phi = 0.35$ and 0.25, respectively. The respective latest updated values of $\mathbf{g}^1$ and $\mathbf{g}^2$ are defined as functions of ϕ.

could alter n_t such that it represents exact time units above a given, integer valued capacity threshold). The method AWT (Section 5.1) identifies and includes all nondominated points in the set `NDP` (for details, see Steuer and Choo, 1983) provided that the parameters $\boldsymbol{\alpha}$ and λ are set appropriately. The while-loop ends when at least one side of the rectangle $B(\mathbf{g}^1, \mathbf{g}^2)$ is below a certain limit (i.e., $g_1^2 - g_1^1 \leq \psi_1$ or $g_2^1 - g_2^2 \leq \psi_2$).

Harvesting efficient solutions. Algorithm 2 only searches for yet-unknown nondominated points in the interior of $B(\mathbf{g}^{\mathrm{TOP}}, \mathbf{g}^{\mathrm{BOT}})$. To initialize the algorithm, the solution $(\bar{\mathbf{x}}, \bar{\mathbf{s}}, \bar{\mathbf{n}}, \hat{\mathbf{z}})$ from the heuristic (Algorithm 1) is provided as input to computing $\mathbf{g}^{\mathrm{TOP}} = \mathbf{g}(\mathbf{y}^{\mathrm{TOP}})$, as defined in (5a). The solution $\mathbf{y}^{\mathrm{TOP}}$ is then provided as input to computing $\mathbf{g}^{\mathrm{BOT}} = \mathbf{g}(\mathbf{y}^{\mathrm{BOT}})$, as defined in (5b). Setting $\mathbf{y}^2 := \mathbf{y}^{\mathrm{BOT}}$, the solution $\mathbf{y}^2$ is used as a starting feasible solution for solving the model TOP and computing $\mathbf{g}^1 = \mathbf{g}(\mathbf{y}^1)$; then the solution $\mathbf{y}^1$ is used as a starting feasible solution for solving the model BOT and computing $\mathbf{g}^2 = \mathbf{g}(\mathbf{y}^2)$. This procedure is repeated, providing updates of the nondominated points $\mathbf{g}^1$ and $\mathbf{g}^2$ and the corresponding solutions $\mathbf{y}^1$ and $\mathbf{y}^2$; the former are included in the set `NDP` of efficient solutions, while the latter are provided as initial feasible solution to the solver.

6. Computational details, tests, and results

We generate 60 instances expected to sufficiently capture most of the realizations of actual data that the model might encounter at GKN Aerospace. All the computations are performed in `Python 3.7` using `Gurobi 9` on a system with 1.70 GHz processor, 4 cores, and 16 GB RAM. Algorithm 2 terminates when at least one of the following criteria is met: (i) the running time for a scalarized problem exceeds 5000 seconds; (ii) the (relative) MIP duality gap is < 0.0005; (iii) MIP absolute gap limit is ≤ 0.01 and there has been no improvements in the duality gap for the previous 1000 nodes of the branch-and-bound tree.[6] Note that optimality is verified (see Algorithm 2) only if the solution satisfies (ii) or (iii).

6.1. Generating the industrial test instances

We have the data for most of the parameters and sets mentioned in Table 2, however, we do not have processing times p_{jk} of jobs $j \in \mathcal{J}$ in machines $k \in \mathcal{N}_j$ (qualification required), and the qualification cost parameters β_{jk}, $j \in \mathcal{J}$, $k \in \mathcal{N}_j$. In order to generate instances that represent possible realizations of the actual data, we introduce the following distributions that are based on knowledge of the managers.

Skewness of processing times. The *skew normal distribution* is a generalized normal distribution allowing for nonzero skewness (Weisstein, 2021).[7] We generate processing times $p_{jk}, k \in \mathcal{N}_j, j \in \mathcal{J}$, for newly qualified machines from three differently skewed normal distributions with mean μ and skewness/shape parameter α: positive skew ($\alpha = 1 > 0$), negative skew ($\alpha = -1 < 0$), and zero skew ($\alpha = 0$). A location parameter/mean μ is based on the expected processing time of a given job type j on an already qualified machine $k \in \mathcal{K}_j \setminus \mathcal{N}_j$ and which is similar to that of the machine being qualified. For all these distributions, we set the scale parameter $\sigma := 0.1 \cdot \mu$; according to the internal statistical process control data (and managerial experience), processing times of newly qualified allocations have a standard deviation of 10% of the expected value.

Qualification cost. The exact cost for qualifying a machine for a job type is not known *a priori* and accurate predictions require detailed simulation work by the engineering team. Thus, the input received are so-called *cost levels*, assigned to each qualification. For testing our model and proposed modifications, we define 20 cost levels, $\mathcal{H} = \{1, \dots, 20\}$, and select the qualification costs from different discrete distributions over the discrete domain $\mathcal{H}$. Letting π_h be the frequency of cost level $h \in \mathcal{H}$, its relative frequency is $\hat{\pi}_h := (\sum_{i\in\mathcal{H}} \pi_i)^{-1}\pi_h$; we also define $\hat{\pi}_0 = 0$. To determine a cost β_{jk}, a sample α is drawn from the interval [0,1]. Then,

$$\beta_{jk} := \begin{cases} h \in \mathcal{H} : \sum_{i=0}^{h-1} \hat{\pi}_i \le \alpha < \sum_{i=0}^{h} \hat{\pi}_i, & \alpha \in [0,1), \\ |\mathcal{H}|, & \alpha = 1. \end{cases}$$

The frequency distributions are defined as follows. For each $h \in \mathcal{H}$, $\pi_h = 1$ (`Uniform`), $\pi_h = h$ (`Right`), $\pi_h = |\mathcal{H}| - (h-1)$ (`Left`), $\pi_h = \min\{\ h\ ;\ |\mathcal{H}| - (h-1)\ \}$ (`Symmetric`), and $\pi_h = \min\{\ h\ ;\ |\mathcal{H}| - (h-1)\ ;\ \max\{h - \lceil\frac{|\mathcal{H}|-1}{2}\rceil\ ;\ \lfloor\frac{|\mathcal{H}|+1}{2}\rfloor - (h-1)\}\}$ (`Bimodal`).

[6]Criterion (iii) is particularly useful while minimizing g_1 as the corresponding lower bounds are often close to 0.

[7]Given the probability density function (pdf) of the standard normal distribution $\phi(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$ and the cumulative distribution function $\Phi(x) = \int_{-\infty}^{x} \phi(t)dt$, the pdf of the skew normal distribution with skewness/shape parameter α is given by $f(x) = 2\phi(x)\Phi(\alpha x)$. The transformation to standard form from the location μ and scale σ parameters is given by $x \to \frac{x-\mu}{\sigma}$.

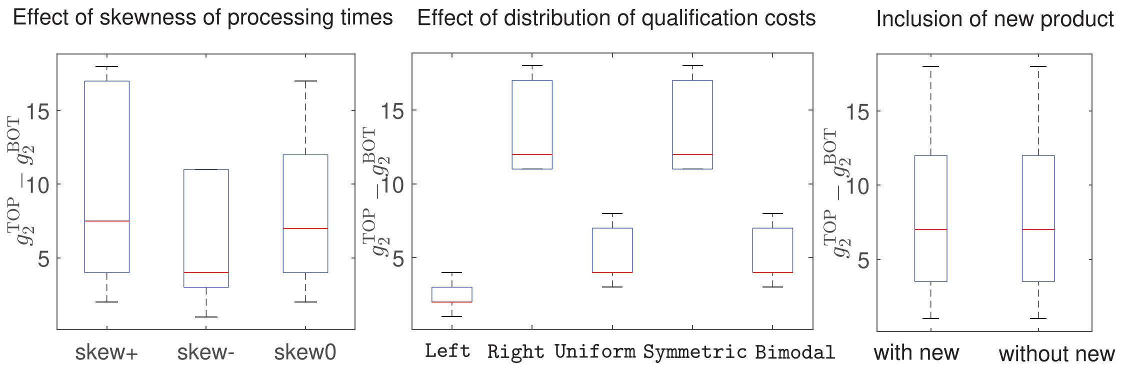

Fig. 2. Distribution of the range for g_2, which affects the maximum number of nondominated points, for different instances.

6.2. Constant data across 60 instances

The demand is from quarterly forecasts made at GKN Aerospace in January 2015 (denoted by J_{15}) and December 2015 (denoted by D_{15}) for the period 2016–2019. The minimum, maximum, and median values of the demand of job types (# of orders) are 1, 172, and 11, respectively. For processing times over the job types and machines, the corresponding values are 0.1, 89.7, and 5.63 hours, respectively. Each machine has a yearly capacity of 5000 hours; the available hours per machine and quarter are thus 1250. The planning period of four years with quarterly time buckets yields $T = 16$ time periods. There are $K = 125$ machines, and $J = 517$ unique job types, each having integral demand per time period. The number of possible assignments of job types to machines during the entire planning period thus amounts to $\sim 10^6$. We employ the parameter values $\tau = 3$, $\gamma = 4$, and $\zeta_k = 0.7, k \in \mathcal{K}$. The instances vary based on (i) distributions used to draw samples of qualification cost parameters (denoted by $\bar{\beta} \in \{\texttt{Left}, \texttt{Right}, \texttt{Symmetric}, \texttt{Uniform}, \texttt{Bimodal}\}$), and processing times (denoted by $\bar{p} \in \{\texttt{skew+}, \texttt{skew-}, \texttt{skew0}\}$), (ii) demand forecast (denoted by $\bar{a} \in \{J_{15}, D_{15}\}$), and (iii) whether a new product is included (with new) or not (without new). Consequently, we have 60 instances. Figure 2 shows box plots for the effect of the above mentioned variations[8] on the range of $g_2(\cdot)$, that is, $g_2^{\text{TOP}} - g_2^{\text{BOT}}$, which also sets the upper limit $g_2^{\text{TOP}} - g_2^{\text{BOT}} + 1$ on the number of nondominated points.

6.3. Results and insights

We compare various state-of-the-art solution approaches with our proposed approach. For each approach, the objectives are combined using augmentation (Aug) or lexicographically (Lex). The criterion space search method used is either the bi-directional ϵ-constraint (BD-ϵ) or the balanced box (BB) method. A switch to the AWT method is denoted by AWT; an $\emptyset$ means that there is no such switch. The solution approaches compared are defined by the 3-tuples (Aug,BD-ϵ,AWT)

[8]Demand forecasts are not shown in Fig. 2 as the two corresponding box plots for the range of $g_2(\cdot)$ were identical.

Table 3
Results from the methods (Aug,BD-ϵ,AWT) (i.e., $\phi = 0.25$) and (Aug,BD-ϵ, $\emptyset$) (i.e., $\phi = 0$) for 60 instances

Distributions		Forecast	With new product				Without new product			
				Solution time [s]				Solution time [s]		
$\bar{\beta}$	$\bar{p}$	$\bar{a}$	Instance	$\phi = 0$	$\phi = 0.25$	|NDP|	Instance	$\phi = 0$	$\phi = 0.25$	|NDP|
Left	skew+	J_{15}	1y	571	–	4	1n	446	–	4
Left	skew-	J_{15}	2y	528	–	3	2n	259	–	4
Left	skew0	J_{15}	3y	570	–	3	3n	357	–	3
Right	skew+	J_{15}	4y	3334	2032	6	4n	2556	1556	6
Right	skew-	J_{15}	5y	2020	2905	6	5n	2238	2280	6
Right	skew0	J_{15}	6y	2414	–	8	6n	2106	–	6
Uniform	skew+	J_{15}	7y	1214	–	5	7n	1159	–	4
Uniform	skew-	J_{15}	8y	1300	933	5	8n	1123	1002	5
Uniform	skew0	J_{15}	9y	744	–	5	9n	744	–	3
Symmetric	skew+	J_{15}	10y	2745	2044	6	10n	2046	–	6
Symmetric	skew-	J_{15}	11y	2043	1942	6	11n	2245	–	6
Symmetric	skew0	J_{15}	12y	2383	–	8	12n	2102	–	5
Bimodal	skew+	J_{15}	13y	1233	–	5	13n	1128	1345	4
Bimodal	skew-	J_{15}	14y	1165	–	5	14n	1009	–	5
Bimodal	skew0	J_{15}	15y	752	–	4	15n	732	–	3
Left	skew+	D_{15}	16y	849	–	4	16n	287	–	3
Left	skew-	D_{15}	17y	0	–	2	17n	0	–	2
Left	skew0	D_{15}	18y	721	–	4	18n	298	–	3
Right	skew+	D_{15}	19y	3762	2556	6	19n	6071	2285	6
Right	skew-	D_{15}	20y	5090	–	6	20n	4018	–	6
Right	skew0	D_{15}	21y	4652	2347	6	21n	6280	2010	4
Uniform	skew+	D_{15}	22y	1264	1380	6	22n	1209	1210	5
Uniform	skew-	D_{15}	23y	1452	–	4	23n	988	–	4
Uniform	skew0	D_{15}	24y	1619	–	4	24n	2772	2255	4
Symmetric	skew+	D_{15}	25y	3913	2662	6	25n	11720	2277	6
Symmetric	skew-	D_{15}	26y	4958	–	6	26n	4222	–	6
Symmetric	skew0	D_{15}	27y	4960	2249	6	27n	6325	2008	4
Bimodal	skew+	D_{15}	28y	1128	1282	5	28n	1277	1138	5
Bimodal	skew-	D_{15}	29y	1722	1596	4	29n	935	–	4
Bimodal	skew0	D_{15}	30y	1297	–	4	30n	823	–	4

(for the values of $\phi \in \{0.25, 0.35\}$), (Aug,BD-ϵ, $\emptyset$), (Aug,BB,$\emptyset$), and (Lex,BD-ϵ,AWT), in total five variants.

Performance. Table 3 compares solution times and provides numbers of nondominated points[9] |NDP| for the methods (Aug,BD-ϵ,AWT), labeled $\phi = 0.25$, and (Aug,BD-ϵ, $\emptyset$), labeled $\phi = 0$. Each row consists of two instances having equal distributions and forecasts, one with new product and one without. Solution times refer to total wall clock time used to find the nondominated

[9]Since we employ a time limit for solving each scalarized problem, it is not guaranteed that all nondominated points are found. However, all methods compared in Fig. 3 found the same nondominated points.

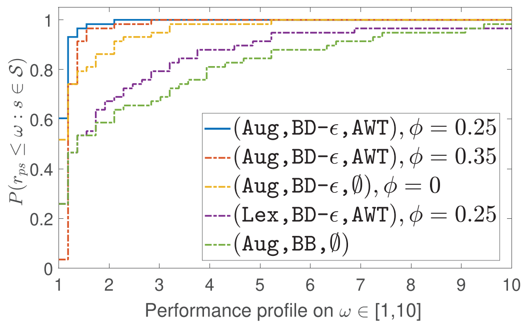

Fig. 3. Performance profiles of the different solution methods while exploring the strict interior of $B(\mathbf{g}^{\mathrm{TOP}}, \mathbf{g}^{\mathrm{BOT}})$.

points—excluding the computation of $\mathbf{g}^{\mathrm{TOP}}$ and $\mathbf{g}^{\mathrm{BOT}}$.[10] Comparisons for solution times should be made between columns labeled as $\phi = 0.25$ and $\phi = 0$, and for the same instance. A "–" indicates that the solution times for (Aug,BD-ϵ,AWT) and (Aug,BD-ϵ, $\emptyset$) are equal, since the switch to the AWT method did not occur. There are 23 instances for which the switch to the AWT method did occur. Six of these 23 instances have either moderately shorter or equal solution times for (Aug,BD-ϵ, $\emptyset$) as compared to (Aug,BD-ϵ,AWT); for the remaining instances, the (Aug,BD-ϵ,AWT) method is computationally superior.

Figure 3 compares the five variants of solution methods mentioned above, as applied to the 60 instances by means of performance profiles (see Dolan and Moré, 2002), with the performance ratio defined as $r_{ps} := \frac{t_{ps}}{\min_{r\in\mathcal{S}}\{t_{pr}\}}$, and t_{ps} denoting the time used by the method $s \in \mathcal{S}$ for solving instance $p \in \mathcal{P}$. The probability that method s solves each of the instances within the ratio ω of the time spent by the fastest method for that instance (i.e., the cumulative distribution function for the performance ratio of method s) is defined as

$$P_s(\omega) := |\mathcal{P}|^{-1}\left|\left\{p \in \mathcal{P} : r_{ps} \le \omega\right\}\right|, \qquad \omega \in [1, 10], \quad s \in \mathcal{S}. \tag{14}$$

The method (Aug,BB,$\emptyset$) performs quite badly for our class of problems. One explanation is that BB solves at most $3|Y_{\mathrm{eff}}|$ scalarized problems (Boland et al., 2015a, Proposition 5), while BD-ϵ solves at most $2|Y_{\mathrm{eff}}| + 1$ scalarized problems (see Chankong and Haimes, 1983), where Y_{eff} is the set of nondominated points. Figure 4 presents a performance profile to highlight the effect of using the starting heuristic (Algorithm 1) within the computation of $\mathbf{g}^{\mathrm{TOP}}$, which along with the corresponding solution $\mathbf{y}^{\mathrm{TOP}}$ is the input to Algorithm 2 (i.e., (Aug,BD-ϵ,AWT) with $\phi = 0.25$). Additionally, we compare the performance of our algorithm with NSGA-II, implemented using the python library pymoo (described in Blank and Deb, 2020). For all the instances, our algorithm resulted in larger *hypervolume* (as defined in Zitzler et al., 2003) than NSGA-II that implies that our algorithm yields a better approximation of the efficient frontier. The ratio of hypervolumes is presented in Fig. 5 for all the instances. Since all the nondominated points were not obtained by NSGA-II, we have not compared it in Fig. 3. For details on parameters used for NSGA-II, we provide access to a jupyter notebook in https://bit.ly/3JKE4DA and corresponding data in our github repository

[10]Instances 17y and 17n have solution time 0, as no NDP exist in the interior of $B(\mathbf{g}^{\mathrm{TOP}}, \mathbf{g}^{\mathrm{BOT}})$, indicated by $g_2^{\mathrm{TOP}} - g_2^{\mathrm{BOT}} = 1$.

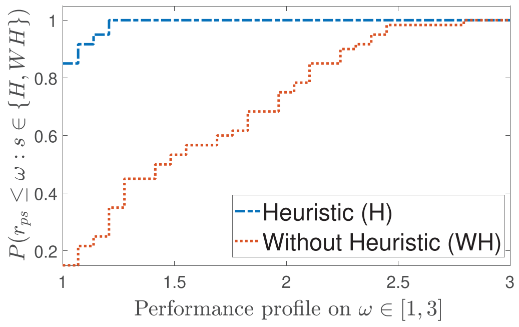

Fig. 4. Performance profiles when $(\bar{\mathbf{x}}, \bar{\mathbf{s}}, \bar{\mathbf{n}}, \hat{\mathbf{z}})$ is provided (H) and when no such solution is provided (WH), while searching for $\mathbf{g}^{\text{TOP}}$.

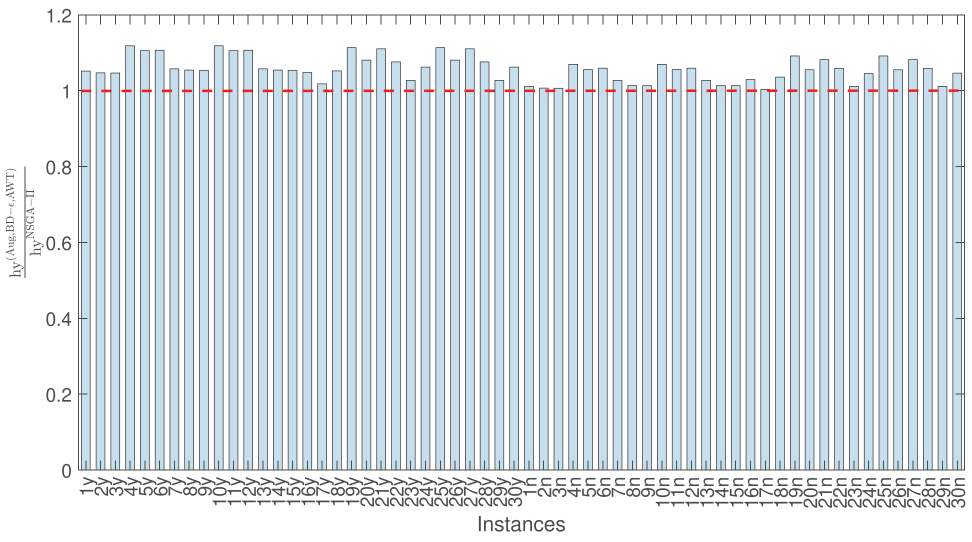

Fig. 5. Ratio of hypervolumes of (Aug,BD-ϵ, AWT), that is, $\text{hy}^{(\text{Aug,BD-}\epsilon,\text{AWT})}$ for $\phi = 0.25$ and hypervolume of NSGA-II, that is, $\text{hy}^{\text{NSGA-II}}$ for all the 60 instances.

https://bit.ly/3iAV4Ao. We have used the feasible solutions obtained from the starting heuristic Algorithm 1 in the initial population and terminated NSGA-II after 3600 seconds.

As a quality check for the starting heuristic, we compare the resulting values for the objectives g_1 and g_2 with $\mathbf{g}^{\text{TOP}}$, which is the nondominated point possessing the minimum possible value for g_1 (see definition in Section 3.3; cf. also the objective in (11)). Figure 6 shows normalized (with respect to $\|\mathbf{g}^{\text{TOP}} - \mathbf{g}^{\text{BOT}}\|_1$) distances between $g_1(\cdot)$ and g_1^{TOP} (where $g_1^{\text{TOP}} = 0$ for all 60 instances), and between $g_2(\cdot)$ and g_2^{TOP}, respectively. The measures indicated are thus given by $\frac{|g_2^{\text{TOP}} - g_2(\bar{\mathbf{x}}, \bar{\mathbf{s}}, \bar{\mathbf{n}}, \hat{\mathbf{z}})|}{g_2^{\text{TOP}} - g_2^{\text{BOT}}}$ and $\frac{g_1(\bar{\mathbf{x}}, \bar{\mathbf{s}}, \bar{\mathbf{n}}, \hat{\mathbf{z}})}{g_1^{\text{BOT}}}$, where $(\bar{\mathbf{x}}, \bar{\mathbf{s}}, \bar{\mathbf{n}}, \hat{\mathbf{z}})$ denotes the solution obtained from Algorithm 1. Note that, out of the 60 instances, the normalized distance equals 0 for six instances of the objective g_1, and for 36 instances of g_2.

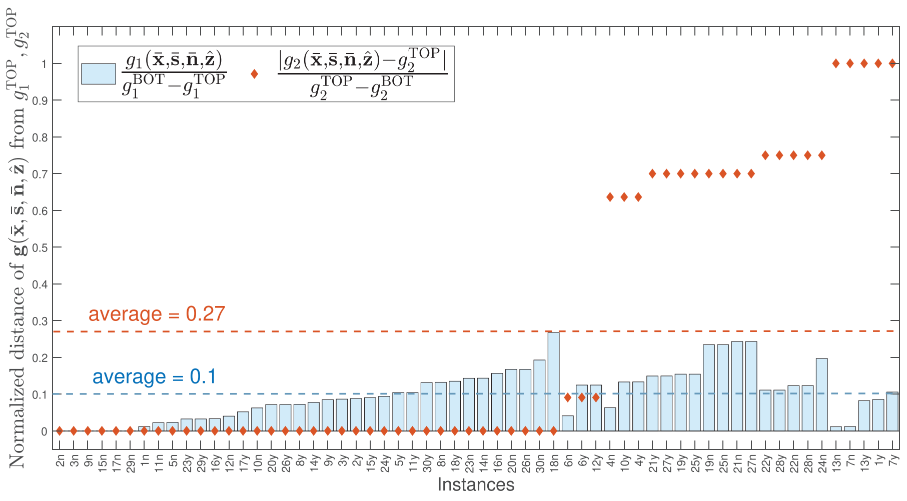

Fig. 6. Normalized distance of the objective values provided by the heuristic, with respect to $\mathbf{g}^{\text{TOP}}$ for 60 instances. The average value of the normalized distances equal 0.1 and 0.27 for g_1 and g_2, respectively. See Section 6.1 for details on instances.

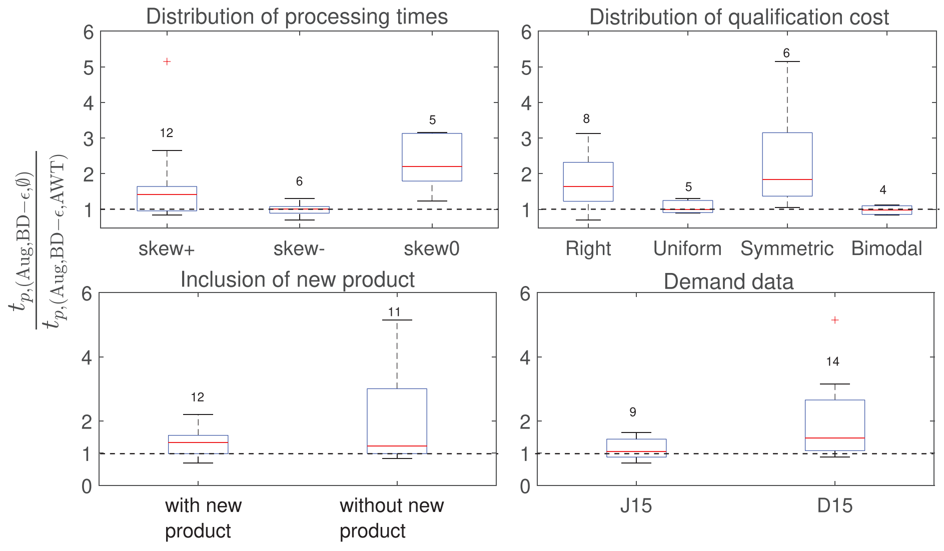

Fig. 7. Ratio of solution times for (Aug,BD-ϵ, $\emptyset$) versus (Aug,BD-ϵ,AWT) for the 23 instances in Table 3 for which the switch to AWT did occur ($\phi = 0.25$). The number of instances in each category is specified above the corresponding box.

Variations among the instances. Figure 7 shows box plots for the ratio of solution times for (Aug,BD-ϵ, $\emptyset$) versus (Aug,BD-ϵ,AWT) (for $\phi = 0.25$). The top-left and top-right plots divide the 23 instances with respect to the skewness of processing times and the distributions of qualification costs, respectively. Furthermore, the bottom-left and bottom-right plots indicate observed variations for new product's inclusion and demand forecast, respectively. It is clear that the instance type has an impact on the magnitude of computational superiority of switching to the AWT. However, for all the 23 instances where AWT is used, we see the ratio is always either close to 1 or significantly greater than 1. It is to be noted that in all tests performed, most of the computation time is spent on improving the lower bounds (verifying optimality). In Fig. A4, we illustrate the efficient frontier for instance $5y$.

6.4. Conclusion

A new resource allocation model for a large tier-1 supplier of aerospace engine systems is presented and analyzed. We have proved various mathematical properties of the tactical resource allocation problem (TRAP) model, one of them resulting in a relaxation of the integrality constraints on some of the variables that is utilized in a specialized heuristic for the TRAP (see Algorithm 1). The solution from the heuristic is used as a starting feasible solution that resulted in significant reductions of solution times (Fig. 4). We tested a tailored *modified bi-directional* ϵ*-constraint* method, which reduces the solution time for a majority of the instances. The algorithm attempts to use the strengths of the ϵ-constraint method while avoiding some of its drawbacks by switching to the augmented weighted Tchebycheff AWT in later stages when the ϵ-constraint is not likely to be tractable. Further tests should be conducted to check the performance of our algorithm as applied to other problem classes.

Future work may involve further polyhedral analysis of the problem aiming at a tighter formulation, and investigating exact decision space search methods, such as branch-and-bound for BOIP. The model (1)–(2) should be extended to a robust formulation, incorporating uncertainty in processing times and qualification costs of newly qualified jobs. To incorporate more preferences of the decision makers, new objectives and/or alternative combinations of the current objectives can be explored as well. Further, we intend to investigate how the inclusion of inventory modeling at different stages of the production planning affects the trade-off between objectives.

Acknowledgments

This work is financially supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA; project no 2017-04845), Chalmers University of Technology, and GKN Aerospace Sweden AB.

References

Ammar, H.B., Ayadi, O., Masmoudi, F., 2019. An effective multi-objective particle swarm optimization for the multi-item capacitated lot-sizing problem with set-up times and backlogging. *Engineering Optimization* 52, 7, 1198–1224.

Aneja, Y.P., Nair, K.P.K., 1979. Bicriteria transportation problem. *Management Science* 25, 1, 73–78.

Blank, J., Deb, K., 2020. Pymoo: multi-objective optimization in python. *IEEE Access* 8, 89497–89509.

Boland, N., Charkhgard, H., Savelsbergh, M., 2015a. A criterion space search algorithm for biobjective integer programming: the balanced box method. *INFORMS Journal on Computing* 27, 4, 735–754.

Boland, N., Charkhgard, H., Savelsbergh, M., 2015b. A criterion space search algorithm for biobjective mixed integer programming: the triangle splitting method. *INFORMS Journal on Computing* 27, 4, 597–618.

Bowman, V.J., 1976. On the relationship of the Tchebycheff norm and the efficient frontier of multiple-criteria objectives. In Thiriez, H., Zionts, S. (eds) *Multiple Criteria Decision Making*, Springer, Berlin, pp. 76–86.

Bradley, J.R., Glynn, P.W., 2002. Managing capacity and inventory jointly in manufacturing systems. *Management Science* 48, 2, 273–288.

Chalmet, L., Lemonidis, L., Elzinga, D., 1986. An algorithm for the bi-criterion integer programming problem. *European Journal of Operational Research* 25, 2, 292–300.

Chankong, V., Haimes, Y., 1983. *Multiobjective decision making: theory and methodology*. Elsevier Science, New York, NY.

Chen, S.P., Huang, W.L., 2010. A membership function approach for aggregate production planning problems in fuzzy environments. *International Journal of Production Research* 48, 23, 7003–7023.

Conforti, M., Cornuéjols, G., Zambelli, G., 2014. *Integer and Combinatorial Optimization*. Springer International Publishing, New York, NY.

Dai, R., Charkhgard, H., 2018. A two-stage approach for bi-objective integer linear programming. *Operations Research Letters* 46, 1, 81–87.

Dächert, K., Gorski, J., Klamroth, K., 2012. An augmented weighted Tchebycheff method with adaptively chosen parameters for discrete bicriteria optimization problems. *Computers & Operations Research* 39, 12, 2929–2943.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2, 182–197.

Díaz-Madroñero, M., Mula, J., Peidro, D., 2014. A review of discrete-time optimization models for tactical production planning. *International Journal of Production Research* 52, 17, 5171–5205.

Dolan, E.D., Moré, J.J., 2002. Benchmarking optimization software with performance profiles. *Mathematical Programming* 91, 2, 201–213.

Ehrgott, M., 2005. *Multicriteria Optimization*. Springer-Verlag, Berlin.

Ehrgott, M., 2006. A discussion of scalarization techniques for multiple objective integer programming. *Annals of Operations Research* 147, 1, 343–360.

Ehrgott, M., Ryan, D.M., 2002. Constructing robust crew schedules with bicriteria optimization. *Journal of Multi-Criteria Decision Analysis* 11, 3, 139–150.

Fleischmann, B., Meyr, H., 1997. The general lotsizing and scheduling problem. *OR Spectrum* 19, 1, 11–21.

Garey, M.R., Johnson, D.S., 1979. *A Guide to the Theory of NP-completeness*. W. H. Freeman & Co, New York, NY.

Genin, P., Lamouri, S., Thomas, A., 2008. Multi-facilities tactical planning robustness with experimental design. *Production Planning & Control* 19, 2, 171–182.

Gupta, A., Maranas, C.D., 1999. A hierarchical Lagrangean relaxation procedure for solving midterm planning problems. *Industrial & Engineering Chemistry Research* 38, 5, 1937–1947.

Hamacher, H.W., Pedersen, C.R., Ruzika, S., 2007. Finding representative systems for discrete bicriterion optimization problems. *Operations Research Letters* 35, 3, 336–344.

Blackstone Jr., J.H., 2013. *APICS Dictionary* (14th edn). Association for Supply Chain Management, Chicago, IL.

Lan, Y., Zhao, R., Tang, W., 2011. Minimum risk criterion for uncertain production planning problems. *Computers & Industrial Engineering* 61, 3, 591–599.

Lasdon, L.S., Terjung, R.C., 1971. An efficient algorithm for multi-item scheduling. *Operations Research* 19, 4, 946–969.

Lawler, E.L., Lenstra, J.K., Kan, A.H.R., Shmoys, D.B., 1993. *Logistics of Production and Inventory*. Elsevier, Amsterdam.

Leitner, M., Ljubić, I., Sinnl, M., Werner, A., 2016. ILP heuristics and a new exact method for bi-objective 0/1 ILPs: application to FTTx-network design. *Computers & Operations Research* 72, 128–146.

Mavrotas, G., 2009. Effective implementation of the ϵ-constraint method in multi-objective mathematical programming problems. *Applied Mathematics and Computation* 213, 2, 455–465.

Mieghem, J.A.V., 2003. Capacity management, investment, and hedging: review and recent developments. *Manufacturing & Service Operations Management* 5, 4, 269–302.

Miettinen, K., 1988. *Nonlinear multiobjective optimization*. Springer Science Business Media, New York.

Miettinen, K., Mäkelä, M.M., Kaario, K., 2006. Experiments with classification-based scalarizing functions in interactive multiobjective optimization. *European Journal of Operational Research* 175, 2, 931–947.

Min, H., Zhou, G., 2002. Supply chain modeling: past, present and future. *Computers & Industrial Engineering* 43, 1–2, 231–249.

Myrelid, A., Olhager, J., 2019. Hybrid manufacturing accounting in mixed process environments: A methodology and a case study. *International Journal of Production Economics* 210, 137–144.

Nourelfath, M., 2011. Service level robustness in stochastic production planning under random machine breakdowns. *European Journal of Operational Research* 212, 1, 81–88.

Parker, M., Ryan, J., 1996. Finding the minimum weight IIS cover of an infeasible system of linear inequalities. *Annals of Mathematics and Artificial Intelligence* 17, 1, 107–126.

Plossl, G.W., Orlicky, J., 1999. *Orlicky's Material Requirements Planning* (2nd edn). McGraw-Hill Professional, New York, NY.

Pochet, Y., Wolsey, L.A., 2006. *Production Planning by Mixed Integer Programming*. Springer, New York, NY.

Ralphs, T.K., Saltzman, M.J., Wiecek, M.M., 2006. An improved algorithm for solving biobjective integer programs. *Annals of Operations Research* 147, 1, 43–70.

Rezaei, J., Davoodi, M., 2011. Multi-objective models for lot-sizing with supplier selection. *International Journal of Production Economics* 130, 1, 77–86.

Steuer, R.E., Choo, E.U., 1983. An interactive weighted Tchebycheff procedure for multiple objective programming. *Mathematical Programming* 26, 3, 326–344.

Talbi, E.G., Basseur, M., Nebro, A.J., Alba, E., 2012. Multi-objective optimization using metaheuristics: non-standard algorithms. *International Transactions in Operational Research* 19, 1-2, 283–305.

Transchel, S., Minner, S., Kallrath, J., Löhndorf, N., Eberhard, U., 2011. A hybrid general lot-sizing and scheduling formulation for a production process with a two-stage product structure. *International Journal of Production Research* 49, 9, 2463–2480.

Ulungu, E., Teghem, J., 1994. The two-phases method: An efficient procedure to solve bi-objective combinatorial optimization problems. *Foundations of Computing and Decision Sciences* 20, 2, 149–165.

Vincent, T., Seipp, F., Ruzika, S., Przybylski, A., Gandibleux, X., 2013. Multiple objective branch and bound for mixed 0–1 linear programming: Corrections and improvements for the biobjective case. *Computers & Operations Research* 40, 1, 498–509.

Wei, C., Li, Y., Cai, X., 2011. Robust optimal policies of production and inventory with uncertain returns and demand. *International Journal of Production Economics* 134, 2, 357–367.

Weisstein, E.W., 2021. Skew normal distribution from MathWorld—a Wolfram web resource. Available at https://reference.wolfram.com/language/ref/SkewNormalDistribution.html (accessed 10 October 2021).

Wörbelauer, M., Meyr, H., Almada-Lobo, B., 2018. Simultaneous lotsizing and scheduling considering secondary resources: a general model, literature review and classification. *OR Spectrum* 41, 1, 1–43.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G., 2003. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation* 7, 2, 117–132.

Özlen, M., Azizoğlu, M., 2009. Multi-objective integer programming: A general approach for generating all non-dominated solutions. *European Journal of Operational Research* 199, 1, 25–35.

Appendix

A.1. Comparison of branch–and–bound trees for the AWT and BD-ϵ or ϵ-constraint scalarized problems

Consider a BOIP with two constraints, two integer variables, and objective functions $g_i := x_i$, $i = 1, 2$:

$$\min_{\mathbf{x}\in\mathbb{Z}_+^2} \left\{(x_1, x_2) \,\middle|\, 3x_1 + 2x_2 \geq 11,\ 11x_1 + 10x_2 \leq 51,\ x_1 \leq 4,\ x_2 \leq 4\right\}. \tag{A1}$$

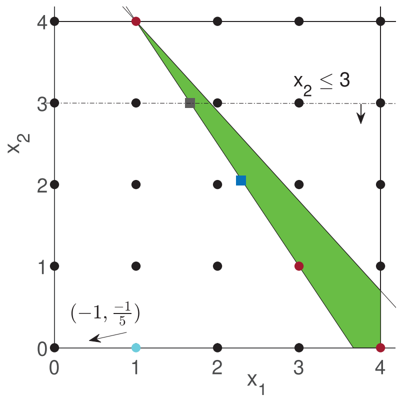

Fig. A1. Feasible region of the LP relaxation of (A1), with $\mathbf{g}^{\text{ref}} = (1, 0)^\top$. Red dots mark nondominated points. Gray and blue squares mark solutions at the root node of (A2) and (A3), respectively.

The points $\mathbf{g}^{\text{TOP}} = (1, 4)^\top$ and $\mathbf{g}^{\text{BOT}} = (4, 0)^\top$ are known and the ideal (reference) point is $\mathbf{g}^{\text{ref}} = (1, 0)^\top$; see Fig. A1. For comparison of the (bi-directional) ϵ-constraint (BD-ϵ) method with the AWT method, we examine the branch–and–bound trees corresponding to their respective scalarized problems. In the BD-ϵ method, the first step is to add the ϵ-constraint $x_2 \le 3$ and solve the scalarized problem

$$\min \left\{ x_1 + \tfrac{1}{5}x_2 \;\middle|\; \mathbf{x} \in X,\ x_2 \le 3 \right\}, \tag{A2}$$

where X is the set defined by the constraints in (A1) and the coefficient $\frac{1}{5}$ is small enough to yield at least a weakly efficient solution. In the AWT method, the first scalarized problem to be solved is defined by the reference point $\mathbf{g}^{\text{ref}} = (1, 0)^\top$, as

$$\min_{f \ge 0, \mathbf{x} \in X} \left\{ f + \lambda(x_1 - 1 + x_2) \;\middle|\; f \ge \alpha_1(x_1 - 1), f \ge \alpha_2 x_2 \right\}, \tag{A3}$$

where the parameters λ, α_1, and α_2 are derived as in (Dächert et al., 2012, Table 2). The branch–and–bound trees for the two scalarized problems in (A2) are shown Figs. A2 and A3.[11] At the root node to (A2), $x_2^* = 3$, such that the ϵ-constraint $x_2 \le 3$ is active, whereas at the root node to (A3), $x_2^* = 2.05$. The conflicting nature of the two objectives typically keeps the ϵ-constraint active for several nodes (in our small example its just once) while solving a scalarization in the BD-ϵ method, while this is not the case for the AWT method.

A.2. The resulting efficient frontier for instance 5y

In Fig. A4, we show nondominated points identified for instance 5y. The two rectangles marked in magenta have area less than $\phi \cdot A(\mathbf{g}^{\text{TOP}}, \mathbf{g}^{\text{BOT}})$, where $\phi = 0.25$, $\mathbf{g}^{\text{BOT}} = (2.25, 17)^\top$ and $\mathbf{g}^{\text{TOP}} = (0, 28)^\top$. Hence, AWT is applied in this instance.

[11]The scalarized problems have an initial feasible solution of $(4, 0)^\top$, used to evaluate upper bounds in Fig. A2.

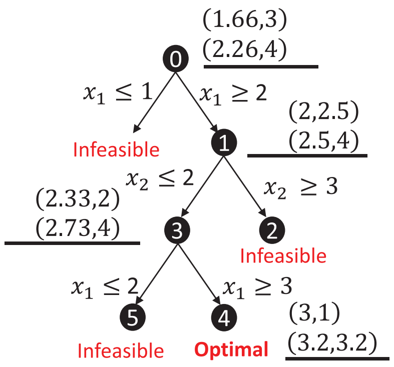

Fig. A2. Branch–and–bound tree for (A2). The two 2-tuples at each node correspond to optimal values of $\mathbf{x} \in \mathbb{R}^2_+$ and (lower bound, upper bound), respectively.

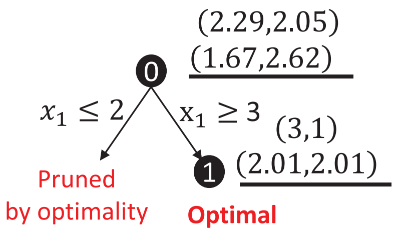

Fig. A3. Branch–and–bound tree for (A3) with $\mathbf{g}^{\text{ref}} = (1, 0)^\top$, $\alpha_1 = 0.612$, $\alpha_2 = 0.388$, and $\lambda = 0.262$.

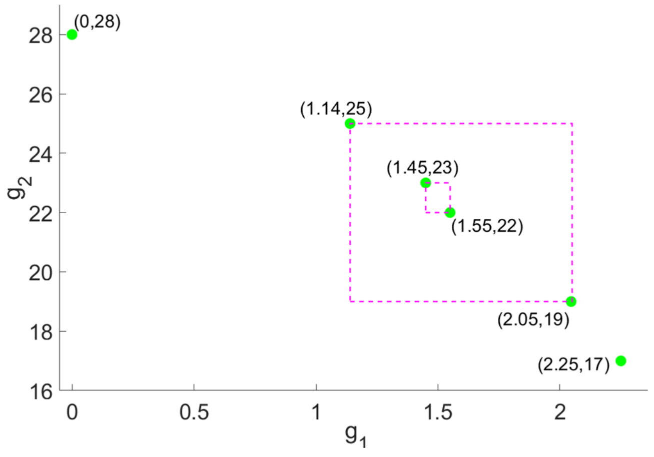

Fig. A4. Nondominated points for instance 5*y*.