Computers & Operations Research 40 (2013) 303–317

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Combining the principles of variable neighborhood decomposition search and the fix&optimize heuristic to solve multi-level lot-sizing and scheduling problems

F. Seeanner *, B. Almada-Lobo, H. Meyr

Fachgebiet Produktion & Supply Chain Management, Technische Universität Darmstadt, Hochschulstr, 1 64 289 Darmstadt, Germany

ARTICLE INFO

Available online 7 July 2012

Keywords:
GLSPMS
Lot-sizing and scheduling
Variable neighborhood decomposition search
VNDS
Exchange
Fix&Optimize

ABSTRACT

In this paper a new heuristic is proposed to solve general multi-level lot-sizing and scheduling problems. The idea is to cross-fertilize the principles of the meta-heuristic Variable Neighborhood Decomposition Search (VNDS) with those of the MIP-based Fix&Optimize heuristic. This combination will make it possible to solve the kind of problems that typically arise in the consumer goods industry due to sequence-dependent setups and shifting bottlenecks. In order to demonstrate the strength of this procedure, a GLSP variant for multiple production stages is chosen as a representative. With the help of artificial and real-world instances, the quality of the solution as well as the computational performance of the new procedure is tested and compared to a standard MIP-solver.

1. Introduction

This paper presents an improvement heuristic based on the principles of the Variable Neighborhood Decomposition Search (VNDS) and Fix&Optimize which is geared to solving multi-level lot-sizing and scheduling problems.

The consumer goods industry is typically characterized by a highly automated flow shop production system which often consists of two or three production stages (e.g., make-and-pack). At each stage several production lines can potentially be used alternatively as they offer – at least partially – the same services. Generally, many final items of different types are produced and they can be assigned to a few setup families. Setup times for changeovers between products of the same family can usually be neglected. In contrast, setup times between different families are significantly sequence-dependent. Therefore, there is a need to simultaneously determine lot-sizes and sequences. Moreover, this industry typically has to face time-varying demands due to seasonality, promotion activities and other factors. As a consequence, the demand mixture of items may change over time. According to the bill-of-materials different product combinations may utilize production stages differently. Therefore, this change can cause so-called "shifting bottlenecks" (on different lines and periods), which enforce a simultaneous consideration of several production levels.

Literature reviews on lot-sizing and scheduling in general are given in [3,5,13,14,28]. Unfortunately, only a few models and solution procedures that meet these requirements do actually exist [4]. One reason for this might be that even single level models are hard to solve in terms of complexity (cf. [5,7,16]). Furthermore, the scalability of solution methods could be a critical issue. Nevertheless, over the past decade, several approaches have been used to solve a variety of capacitated lot-sizing models. Buschkühl et al. [3] roughly classify these approaches in five groups, mathematical programming, Lagrangean relaxation, decomposition, aggregation, and problem specific greedy algorithms as well as meta-heuristics. In particular, members of the last group, such as Simulated Annealing, Tabu Search, or Genetic Algorithms, have been successfully used to solve hard lot-sizing problems (see [13]). Consequently, this paper proposes a procedure which is able to solve multi-level lot-sizing and scheduling problems in a reasonable amount of time. It represents a combination of a meta-heuristic and a heuristic based on an exact mathematical programming approach which resembles the work of James and Almada-Lobo [12]. Strictly speaking, this approach brings together the Variable Neighborhood Decomposition Search and the Fix&Optimize heuristic, also known as "Exchange".

Hansen and Mladenović [9] developed the principle of the Variable Neighborhood Search (VNS), which relies on the concept of neighborhood search. In order to avoid being entrapped in a local optimum, which is not the global one, VNS systematically changes the neighborhood structure in the shaking phase. In recent years, many variants of this meta-heuristic have been successfully applied to a wide range of optimization problems,

* Corresponding author.
E-mail addresses: florian.seeanner@googlemail.com (F. Seeanner), almada.lobo@fe.up.pt (B. Almada-Lobo), h.meyr@uni-hohenheim.de (H. Meyr).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.07.002

such as problems of graph theory (e.g., the traveling salesman problem [6,9], minimum spanning trees [19,22]), supply chain planning problems (e.g., car sequencing [21]), continuous optimization [18] or lot-sizing and scheduling problems [1,2]. One variant of VNS is called the Variable Neighborhood Decomposition Search (VNDS) [10]. In contrast to standard VNS, it does not search the whole solution space, it only searches a subset. This is the result of a kind of decomposition.

In contrast, the Fix&Optimize is a Mixed-Integer-Programming (MIP) based improvement heuristic, which iteratively solves a series of sub-MIPs. It starts with a given solution and decomposes the integer variables into two subsets in every step. Some of the variables are fixed to the values found so far. The other variables, however, are "released" and are to be optimized again [20]. Accordingly, a feasible solution can always be found using a standard MIP-solver and thus, a new solution is at least as good as the old one [20]. It is important to note that it is crucial to decide which and how many variables should be released. Several iterations of this procedure with different subsets are typically executed. In the last couple of years this heuristic has become quite popular. It is referenced as "Fix&Optimize" and this heuristic has been applied to lot-sizing (and scheduling) problems quite successfully [11,23,24,27].

As in the approach by Lazić et al. [15], who use VNDS with variable fixing to solve 0–1 mixed integer programs, the basic idea is to apply the concept of VNDS in order to methodically adapt the variable sets for the Fix&Optimize heuristic.

In order to demonstrate the quality of the proposed procedure the General Lot-sizing and Scheduling Problem for Multiple production Stages (GLSPMS) [17,26] is chosen. It represents one of the first generalized large-time-bucket (LTB) models using a small-time-bucket (STB) sub-structure, which makes it possible to simultaneously lot-size and schedule multiple levels with heterogeneous, parallel production lines per stage and sequence-dependent setup times. The resulting plans of this model provide compact schedules based on lot-streaming without unrealistic lead-times, which are common for LTB-models. Unfortunately, standard MIP solvers like Xpress or Cplex are only able to solve medium-sized instances (e.g., three lines, 12 products, six periods) in a reasonable amount of time (1 h). Therefore, the GLSPMS seems to be a good starting point to demonstrate the strength of our solution procedure.

Section 2 introduces the GLSPMS model. Section 3 outlines how the Fix&Optimize of Helber and Sahling [11] for the MLCLSP is adapted to a "pure" Fix&Optimize approach for the GLSPMS in order to be able to combine it with VNDS. The new solution procedure that results from a combination of VNDS and the adapted Fix&Optimize is then presented and in Section 4 test instances and corresponding parameter settings are presented, followed by computational results. Finally, a short summary and outlook are provided in Section 5.

2. The general lot-sizing and scheduling problem for multiple production stages

The following model is based on the GLSPMS formulation presented by Meyr [17]. Only a few adaptations (concerning constraints (7), (8), (20) and (21)) are made to make the plans even more compact and to reduce the number of variables. These will be explained after the model formulation has been shown. Further reformulations of this model are tested in [26].

It is considered that $j=1,\dots,J$ (physical) products have to be scheduled on $l=1,\dots,L$ production lines over a finite planning horizon. Common LTB-models divide this planning horizon into $t=1,\dots,T$ non-overlapping large time buckets (e.g., weeks or months) – in the following denoted as macroperiods. The basic idea of the *single-level*, single-line GLSP [8] and its *single-level* adaptation to parallel production lines GLSPPL [16] was that the planning horizon is additionally subdivided into $s=1,\dots,S$ non-overlapping microperiods of variable lengths. This means a production line can only be set up for a single product per microperiod and the model implicitly determines the length of the microperiod by the production quantity produced within. Let $y_{ljs}\in\{0;1\}$ be a binary variable denoting whether a production line l is set up for product j in microperiod s ($y_{ljs}=1$) or not ($y_{ljs}=0$) and $x_{ljs}\geq 0$ be a continuous variable denoting the unknown production quantity of j in s that is produced on l. Given a production coefficient a_{lj} that represents the time to produce one unit of j on l, the length of microperiod s on line l can be computed by $a_{lj}x_{ljs}$ if $y_{ljs}=1$.

Therefore, in the single-level, parallel-line case the "same" microperiod s can have different lengths depending on whether production on line $l=1$ or on line $l=2$ is considered. This is fundamentally different to the multi-level case represented by the GLSPMS. Once again, l denotes the different production lines, but these can also be settled on different production stages.[1] Once again, the lengths of the microperiods s are implicitly determined by the model and are thus variable. However, now a microperiod s has the same length for all production lines. This is necessary to synchronize the material flow between different stages of production. It is important to ensure, for example, that a successor product on a successor line cannot start production before its corresponding predecessor product on the predecessor line has started. This issue will be discussed further when discussing the constraints of the model in more detail.

Fig. 1 illustrates this time structure of the multi-level GLSPMS. Since the microperiods s start at the same point in time for all lines l, it is possible to introduce a continuous variable $w_s\geq 0$ denoting the starting time of microperiod s. The start (or end) of a macroperiod t is known in advance for each production line l. In common LTB-models it is given by the capacity of the production line in this macroperiod. Therefore, additional microperiods $s\in\Phi$ can be introduced, with starting times that are fixed to these points in time. This is shown in the upper and middle part of Fig. 1. The upper part graphically illustrates the setup sequences ($y_{ljs}=1$) per microperiod for each production line. For example, line $l=2$ is set up for product $j=3$ in microperiod $s=1$, for $j=4$ in $s=2$ and for $j=3$ again in $s=3,4$. The starting times of the microperiods $s=1$ and $s=5$ represent the boundaries of a macroperiod and are thus fixed (middle part of the figure). However, the starting times of microperiods $s=2,3$ and 4 can be freely determined by the GLSPMS model. Therefore, the lengths of the microperiods $s=1,\dots,4$ are variable, even though the overall length of the macroperiod is fixed to w_5-w_1.

The potential events within a microperiod will now be considered for a microperiod s and a line l which is set up for a unique product j. As already noted, there can be a production process (expressed by x_{ljs}) generating this product. A changeover to another product could also be necessary, incurring some changeover time. It could also be the case that the line is idle (i.e., it is neither producing nor setting up). Here the line may remain in a sort of "standby mode", which means that the same product j can be produced again without needing a further setup for j. This type of idleness is often called "conservation of the setup state" (see e.g., [8]). Nevertheless time-dependent costs c_l^i for the duration x_{ls}^i of this standby might occur. The two events "production" and "changing over" can be

[1] Note that the assignment of production lines to production stages will only be indirectly captured by the bill-of-material (BOM) coefficients and the production coefficients.