RAIRO-Oper. Res. 55 (2021) 1197–1212
https://doi.org/10.1051/ro/2021049

RAIRO Operations Research
www.rairo-ro.org

THE INTEGRATED UNCAPACITATED LOT SIZING AND BIN PACKING PROBLEM

Natã Goulart[1], Thiago F. Noronha[2], Martin G. Ravetti[2] and Mauricio C. de Souza[3,*]

Abstract. In the integrated uncapacitated lot sizing and bin packing problem, we have to couple lot sizing decisions of replenishment from single product suppliers with bin packing decisions in the delivery of client orders. A client order is composed of quantities of each product, and the quantities of such an order must be delivered all together no later than a given period. The quantities of an order must all be packed in the same bin, and may be delivered in advance if it is advantageous in terms of costs. We assume a large enough set of homogeneous bins available at each period. The costs involved are setup and inventory holding costs and the cost to use a bin as well. All costs are variable in the planning horizon, and the objective is to minimize the total cost incurred. We propose mixed integer linear programming formulations and a combinatorial relaxation where it is no longer necessary to keep track of the specific bin where each order is packed. An aggregate delivering capacity is computed instead. We also propose heuristics using different strategies to couple the lot sizing and the bin packing subproblems. Computational experiments on instances with different configurations showed that the proposed methods are efficient ways to obtain small optimality gaps in reduced computational times.

Mathematics Subject Classification. 90B99, 90C11, 90C27.

Received August 6, 2020. Accepted March 26, 2021.

1. Introduction

In this paper, we deal with the Integrated Uncapacitated Lot Sizing and Bin Packing problem (IULSBP, for short). It has been emphasized in the literature the importance of dealing simultaneously with interdependent decisions to exploit the optimization potential in integrated systems, see [7,16]. We concentrate on the IULSBP as it appears in practical situations where an intermediary company or a warehouse buys in bulk or submits replenishment orders to suppliers of single products to compose client orders. Figure 1 illustrates this situation. There are three suppliers of single products, identified by $p = 1, 2, 3$. A warehouse, represented by a circle in

Keywords. Integrated production-delivering problems, lot sizing, bin packing, Heuristics.

[1] Departamento de Tecnologia em Engenharia Civil Computação e Humanidades, Universidade Federal de São João del-Rei, Rodovia MG 443, KM 7, cep: 36420-000, Ouro Branco, MG, Brazil.

[2] Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Av. Antônio Carlos, 6627, cep: 31270-901, Belo Horizonte, MG, Brazil.

[3] Departamento de Engenharia de Produção, Universidade Federal de Minas Gerais, Av. Antônio Carlos, 6627, cep: 31270-901, Belo Horizonte, MG, Brazil.

*Corresponding author: prof.mauriciodesouza@gmail.com

Article published by EDP Sciences

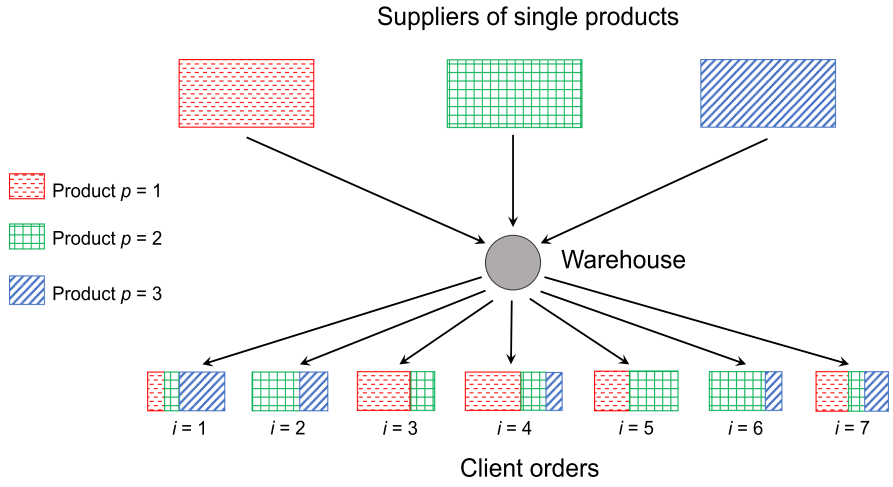

FIGURE 1. Schematic representation of the context of application of the IULSBP.

the center of the figure, receives the products from these suppliers and composes orders with them. An order is composed of quantities of different products to meet client requests. In Figure 1 the warehouse composes and delivers orders for seven clients, identified by $i = 1, \ldots, 7$. We assume the delivery of client orders to be made in containers or by trucks, so the company has to manage on one side lot sizing decisions and on the other side bin packing decisions. An example of such a situation is a company the buys fruits from farmers to meet restaurants' demands, the company places orders of mango, ananas, etc., directly to farmers and composes different orders with these products to be delivered to restaurants. Situations of this kind can found in many sectors of industry and services.

We are given a set T of periods, a set V of client orders, and a set P of products. An order $i \in V$ is composed by quantities q_{ip} of each product $p \in P$ to be delivered all together to a client no later than a period $t_i \in T$. We assume a set K of homogeneous bins with capacity Q available at each period $t \in T$, such that $|K|$ is large enough. The quantities of an order i must all be packed in the same bin $k \in K$. On the other hand, a bin can be used to pack more than one order if its capacity is not exceeded.

Given the sets V, P, T, and K, as described above, the IULSBP consists of jointly deciding over the planning horizon (i) the size of the lots of each product $p \in P$ to buy at each period $t \in T$, (ii) how to pack the orders in V, and (iii) when to deliver each bin. There is a setup cost c_{pt} incurred when a lot of product p is bought at period t, and a holding cost e_{pt} for each unit of product p left in inventory at the end of period t. An order $i \in V$ can be packed in any bin to be delivered at $t = 1, \ldots, t_i$, as long as the quantities q_{ip} of all products composing the order are available. It is worth noting that even if order i is packed in a bin to be delivered in a period $t < t_i$, the inventory holding cost will be incurred for quantities q_{ip} at periods $t, t+1, \ldots, t_i - 1$, since capital is tied up as the order due date is t_i. We assume a cost f_t of delivering a bin at period $t \in T$. The IULSBP problem consists of meeting all the client orders with the minimum total cost. Figure 2 illustrates the integration of lot sizing and bin packing decisions occurring in the IULSBP. The central line in the figure represents the planning horizon divided into time periods. In the upper part of the central line, the lot sizing decisions are represented by incoming supplying replenishments of three products over the time periods. In the bottom part of the central line, the packing decisions are represented by outgoing rectangles in solid line. Each rectangle represents a bin filled with one or more client orders composed of different products. The bins packing client orders are delivered over the time periods.

To the best of our knowledge, few studies in the literature are directly related to the problem studied in this paper. Ben-Khedher and Yano [4] address a problem in which demands for items are given in terms of

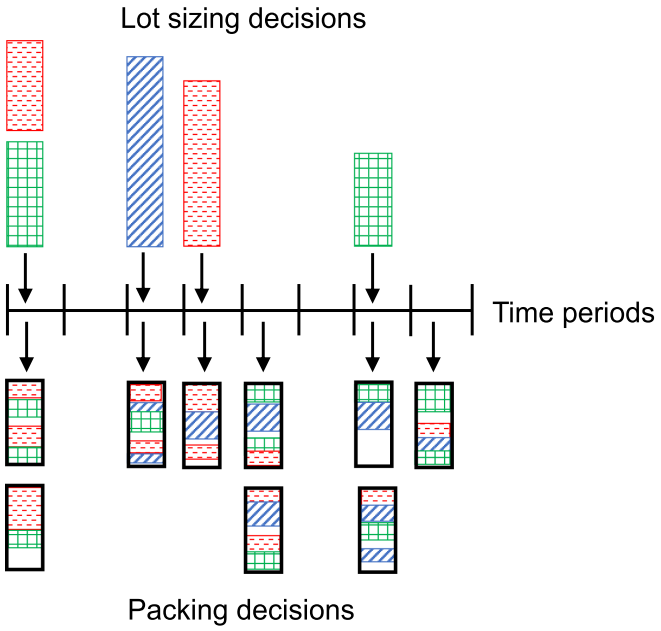

FIGURE 2. Schematic representation of the lot sizing and packing decisions over time.

containers, and the containers must be packed into trucks for shipment. There is no setup cost, so items are bought according to shipping decisions. Molina *et al.* [12] extend the work of Norden and Velde [13] and propose models for production lot sizing problems with distribution costs using unit load devices such as pallets and containers. Lot sizing decisions are subject to capacity constraints, and items must be packed for shipping at the very period they are produced. Also, the cost of delivering a truck is the same in every period. In the problem under study in this paper, the fact that (i) the products are tied in the form of client orders to be delivered as a whole, (ii) the cost of delivering a bin varies from one period to another, and (iii) delivering are not constrained to occur in the very period a product is bought or produced, makes the integration between lot sizing and packing decisions more challenging than in those previous approaches.

A few papers are partially related to this work. Stecke and Zhao [17] study the problem faced by a make-to-order manufacturing company adopting a commit-to-delivery business mode. A third-party company offers different shipping modes, and the goal is to plan lot sizes with transportation costs. Lee *et al.* [9] consider lot sizing subject to a carrying capacity of container constraint. The production orders are shipped by containers in the very period they occur, and the total freight cost is proportional to the number of containers used in an aggregate manner. No packing decisions are involved. Sancak and Salman [15] analyze production plans with minimum transportation and inventory holding costs where transportation costs are given by the number of trucks delivered. Backlog is allowed since a less-than-full truckload shipment can be postponed to the next period. Again, no packing decisions are involved. Melega *et al.* [11] review the literature related to the integration between the lot sizing and the cutting stock problem. The authors classify the literature into two types of integration based on the time and on the production level dimensions. Much more attention has been given in the literature to the integration of lot sizing and vehicle routing. Coelho *et al.* [6] provide a comprehensive review of the inventory routing problem, which combines vehicle routing with inventory management decisions. A more complete version of that problem involves production decisions as well, see [1,3]. More recently, Azadeh *et al.* [2] address the particularity of perishable products in the context of the inventory routing problem. Schmid *et al.* [16] provide an overview on extensions of the vehicle routing with dealing with lot sizing, scheduling, packing, batching, inventory and intermodality.

Concerning the IULSBP problem, we propose, in this paper, a mixed integer linear programming formulation for the problem, and then a reformulation using the so-called multicommodity formulation of the lot sizing variables. Preliminary computational experiments have shown that even with the multicommodity reformulation we were able to solve only small instances within a time limit of one hour. So, the aim of this paper is to develop efficient ways to obtain lower and upper bounds in reduced computational time. To this end, we first propose a combinatorial relaxation to provide good lower bounds with less computational effort. Then, we propose four heuristics to obtain good upper bounds. Computational experiments showed that the proposed combinatorial relaxation was able to provide good lower bounds in reduced computational times. Besides, they showed that the best of the proposed heuristics obtained solutions with average optimally gaps of at most 3.3% within one minute of running time, on average.

The paper is organized as follows. In Section 2, we propose mixed linear integer programming formulations for IULSBP. In Section 3 the combinatorial relaxation to obtain lower bounds. The proposed heuristics are described in Section 4. We report computational experiments in Section 5, and we draw the concluding remarks in the last section.

2. MILP formulations and relaxation for IULSBP

In this section, we first propose a mixed integer linear programming (MILP) formulation for IULSBP, called the Single Commodity Flow formulation (SF). Next, we show a reformulation of SF using the so-called multicommodity formulation of the lot sizing variables, see [14]. We refer to this formulation as the Multicommodity Flow formulation (MF).

2.1. SF formulation

A formulation for IULSBP can be obtained from variables v_{pt}, such that $v_{pt} = 1$ if product $p \in P$ is bought at period $t \in T$ and $v_{pt} = 0$ otherwise; and y^t_{ik}, such that $y^t_{ik} = 1$ if order $i \in V$ is packed in bin $k \in K$ and delivered at period $t \in \{1, \ldots, t_i\}$ and $y^t_{ik} = 0$ otherwise. The following continous variables are also necessary. Variables x_{pt} hold how many units of $p \in P$ are bought in $t \in T$, and variables h_{pt} hold how many units of $p \in P$ are left in inventory at the end of period $t \in T$. Besides, we also use variables z_{kt}, which are naturally 0 or 1, such that $z_{kt} = 1$ if the bin $k \in K$ is delivered at period $t \in T$ and $z_{kt} = 0$ otherwise.

$$\min \sum_{p\in P}\sum_{t\in T} c_{pt}v_{pt} + \sum_{p\in P}\sum_{t\in T} e_{pt}h_{pt} + \sum_{k\in K}\sum_{t\in T} f_t z_{kt} \tag{2.1}$$

$$x_{pt} \leqslant M_{pt}v_{pt} \qquad \forall p \in P,\ \forall t \in T \tag{2.2}$$

$$h_{pt-1} + x_{pt} - h_{pt} = \sum_{i\in V\,:\,t=t_i} q_{ip} \qquad \forall p \in P,\ \forall t \in T \tag{2.3}$$

$$\sum_{u=1}^{t} x_{pu} \geqslant \sum_{u=1}^{t} \sum_{i\in V: u\in\{1,\ldots,t_i\}} q_{ip} \sum_{k\in K} y^u_{ik} \qquad \forall p \in P,\ \forall t \in T \tag{2.4}$$

$$z_{kt} \geqslant y^t_{ik} \qquad \forall k \in K,\ \forall i \in V, t = 1, \ldots, t_i \tag{2.5}$$

$$\sum_{k\in K}\sum_{t=1}^{t_i} y^t_{ik} = 1 \qquad \forall i \in V \tag{2.6}$$

$$\sum_{i\in V\,:\,t\in\{1,\ldots,t_i\}} \sum_{p\in P} q_{ip}y^t_{ik} \leqslant Qz_{kt} \qquad \forall k \in K,\ \forall t \in T \tag{2.7}$$

$$\sum_{i\in V\,:\,t\in\{1,\ldots,t_i\}} y^t_{ik} \leqslant \sum_{i\in V\,:\,t\in\{1,\ldots,t_i\}} y^t_{ik-1} \qquad \forall k \in K,\ \forall t \in T \tag{2.8}$$

$$z_{kt} \geqslant 0 \qquad \forall k \in K,\ \forall t \in T \tag{2.9}$$

$$y_{ik}^t \in \{0,1\} \qquad \forall k \in K,\ \forall i \in V, t = 1,\dots,t_i \tag{2.10}$$
$$v_{pt} \in \{0,1\} \qquad \forall p \in P,\ \forall t \in T \tag{2.11}$$
$$x_{pt} \geqslant 0 \qquad \forall p \in P,\ \forall t \in T \tag{2.12}$$
$$h_{pt} \geqslant 0 \qquad \forall p \in P,\ \forall t \in T. \tag{2.13}$$

The resulting formulation, called the single commodity flow formulation (SF), is defined by (2.1)–(2.13). Given an instance of IULSBP characterized by the tuple $\langle V, P, T, K, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t\rangle$, the objective function (2.1) minimizes the sum of the setup costs, inventory holding costs, and delivering costs, respectively. The constraints in (2.2) enforce that a cost v_{pt} is charged if any number of units of product $p \in P$ is bought at period $t \in T$, where the constant $M_{pt} = \sum_{i \in V : t_i \geqslant t} q_{ip}$. The inventory balance constraints are enforced by (2.3). The constraints in (2.4) guarantee that the quantities of each product $p \in P$ to fulfill delivery decisions at each period $t \in T$ are available. The constraints in (2.5) require that a cost f_t is charged if a bin $k \in K$ is used to deliver an order $i \in V$ at a period $t \in T$. The constraint that an order $i \in V$ should be packed in a single bin is enforced by (2.6), while the bin capacity constraints are imposed by (2.7). The constraints (2.8) break symmetry, reducing the number of equivalent solutions in the search space. Finally, the constraints in (2.9)–(2.13) define the domain of the variables z_{kt}, y_{ik}^t, v_{pt}, x_{pt} and h_{pt}, respectively.

2.2. MF reformulation

In this section, we present a reformulation of SF exploiting the fact that the convex hull of the uncapacitated lot sizing problem can be described with a multicommodity formulation, see [14]. To do this, the inventory variables h_{pt}, for all $p \in P$ and $t \in T$, are suppressed and constraints (2.3) and objective function (2.1) are rewritten as follows

$$h_{pt} = \sum_{u=1}^{t} x_{pu} - \sum_{u=1}^{t} \sum_{i \in V\,:\,u=t_i} q_{ip}$$
$$\min \sum_{p \in P} \sum_{t \in T} c_{pt} v_{pt} + \sum_{p \in P} \sum_{t \in T} \bar{e}_{pt} x_{pt} + \sum_{k \in K} \sum_{t \in T} f_t z_{kt}$$

where $\bar{e}_{pt} = \sum_{u \in T : u \geqslant t} e_{pu}$, and adding the constant term $-\sum_{p \in P} \sum_{t \in T} e_{pt} \sum_{u=1}^{t} \sum_{i \in V\,:\,u=t_i} q_{ip}$.

We then replace variables x_{pt} by variables $x_{\tau t}^{ip}$ corresponding to the number of units of product $p \in P$ bought at period $\tau \in \{1,\dots,t_i\}$ to be delivered at period $t \in \{\tau,\dots,t_i\}$ to meet order $i \in V$. The resulting MF formulation is defined by the following objective function (2.14) and constraints (2.15)–(2.17), as well as the constraints (2.5)–(2.11) from the previous formulation. Our aim is to improve the lower bounds provided by the linear relaxation of SF.

$$\min \sum_{p \in P} \sum_{t \in T} c_{pt} v_{pt} + \sum_{p \in P} \sum_{\tau \in T} \bar{e}_{p\tau} \sum_{i \in V\,:\,\tau \in \{1,\dots,t_i\}} \sum_{t=\tau}^{t_i} x_{\tau t}^{ip} + \sum_{k \in K} \sum_{t \in T} f_t z_{kt} \tag{2.14}$$

$$x_{\tau t}^{ip} \leqslant q_{ip} v_{p\tau} \qquad \forall p \in P,\ \forall i \in V, t = 1,\dots,t_i, \tau = 1,\dots,t \tag{2.15}$$
$$\sum_{\tau=1}^{t} x_{\tau t}^{ip} = q_{ip} \sum_{k \in K} y_{ik}^t \qquad \forall i \in V,\ \forall p \in P, t = 1,\dots,t_i \tag{2.16}$$
$$x_{\tau t}^{ip} \geqslant 0 \qquad \forall p \in P,\ \forall i \in V, t = 1,\dots,t_i, \tau = 1,\dots,t. \tag{2.17}$$

The objective function (2.14) minimizes the sum of the setup costs, inventory holding costs, and delivering costs, respectively. The constraints in (2.15) enforce that a setup cost $v_{p\tau}$ is charged if any number of units of product $p \in P$ is bought at period $\tau \in \{1,\dots,t\}$. The constraints in (2.16) ensure that in case of order $i \in V$

is served at period $t \in \{1, \ldots, t_i\}$, the number of units of product $p \in P$, bought at the periods from 1 to t, to meet this order, is equal its demand for this product. Finally, the constraints in (2.17) define the domain of the variables $x_{\tau t}^{ip}$.

3. Combinatorial Relaxation of SF

We propose a Combinatorial Relaxation (CR) of SF to obtain good lower bounds with less computational effort. The main point is that the constraints due to packing the orders to be delivered in a period into bins are relaxed in an aggregate capacity manner. In CR an order has still to be delivered entirely in a single period, but the products corresponding to an order no longer need to be packed all together in a single bin. Instead, an aggregated capacity is assigned to each period. Thus, it is no longer necessary to keep track of the specific bin where each order is packed. The resulting formulation is obtained by removing variables z_{kt} and replacing the binary variables y_{ik}^t, for all $k \in K$, $i \in V$ and $t \in \{1, \ldots, t_i\}$, by the variables y_{it}, such that $y_{it} = 1$ if order $i \in V$ is delivered at period $t \in \{1, \ldots, t_i\}$ and $y_{it} = 0$ otherwise. Besides, we add the variables $w_t \geqslant 0$ to obtain an aggregated capacity in terms of Q in each period $t \in T$. The resulting $\text{CR}^{w_t \geqslant 0}$ formulation is defined by (3.1)–(3.6) and the constraints (2.2), (2.3), and (2.11)–(2.13) from SF.

$$\min \sum_{p \in P} \sum_{t \in T} c_{pt} v_{pt} + \sum_{p \in P} \sum_{t \in T} e_{pt} h_{pt} + \sum_{t \in T} f_t w_t \tag{3.1}$$

$$\sum_{u=1}^{t} x_{pu} \geqslant \sum_{u=1}^{t} \sum_{i \in V : u \in \{1, \ldots, t_i\}} q_{ip} y_{iu} \qquad \forall p \in P,\ \forall t \in T \tag{3.2}$$

$$\sum_{t=1}^{t_i} y_{it} = 1 \qquad \forall i \in V \tag{3.3}$$

$$\sum_{i \in V\,:\, t \in \{1, \ldots, t_i\}} \sum_{p \in P} q_{ip} y_{it} \leqslant Q w_t \qquad \forall t \in T \tag{3.4}$$

$$y_{it} \in \{0, 1\} \qquad \forall i \in V, t = 1, \ldots, t_i \tag{3.5}$$

$$w_t \geqslant 0 \qquad \forall t \in T. \tag{3.6}$$

The object function (3.1) minimizes the sum of the setup costs, inventory holding costs, and delivering costs, respectively. The constraints in (3.2) guarantee that the quantities of each product $p \in P$ to fulfill delivery decisions at each period $t \in T$ are available. The constraints that an order $i \in V$ should be delivered in a single period is imposed by (3.3), while the correct value of w_t is enforced by (3.4). Finally, the constraints in (3.5) and (3.6) define the domain of the variables y_{it} and w_t, respectively.

4. Heuristics for IULSBP

We propose four heuristics for IULSBP. Each one is based on a different approach to decouple IULSBP into lot sizing and the bin packing subproblems. The lot sizing subproblems are solved independently for each product $p \in P$ with the algorithm WW of Wagner and Whitin [19], while the bin packing subproblems are solved with the VPSolver of Brandão and Pedroso [5], which is a stated-of-the-art exact algorithm for the bin packing problem. We aim at identifying which of these approaches is the most efficient to find good solutions (upper bounds) for IULSBP, in order to guide future studies of heuristics for this problem.

Given an instance characterized by the tuple $\langle V, P, T, K, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t \rangle$ the heuristics return a solution characterized by tuple $\langle T_p, S \rangle$. The subset $T_p \subseteq T$ represents the periods in which there are purchases of $p \in P$, and S is a set of tuples $\langle b, t \rangle$, where $b \subseteq V$ is a subset of orders that are packed in the same bin and delivered at the same period $t \in T$. The value of how many units of product $p \in P$ are bought at period $t \in T$ can be directly computed from $\langle T_p, S \rangle$.

4.1. Uncoupled Heuristic (UH)

The UH heuristic solves the lot sizing and the bin packing subproblems independently. The orders are always delivered at their due date, so the solution of the bin packing subproblem has no impact on the lot sizing subproblem, and *vice versa*. No effort is made to combine the solutions of both subproblems to find better solutions for the integrated problem. It is included in this paper so one can assess how good are the solutions of IULSBP compared with an approach that disregards the relationship between lot sizing and bin packing decisions.

The pseudo-code of UH is described in Algorithm 1. An instance of the lot sizing problem for each $p \in P$, where demands in each period $t \in T$ are given by $\sum_{i \in V: t_i = t} q_{ip}$, is solved in line 1. The set S is initialized in line 2. In the loop from lines 3 to 7, a bin packing instance with the subset $V_t \subseteq V$ of orders i whose due date $t_i = t$ is solved for each $t \in T$. The resulting bin packing solution is stored in $B_t \subset 2^{V_t}$, and the for loop in lines 6 and 7 inserts each bin $b \in B_t$ in the solution S. UH stops in line 8, where the solution $\langle T_p, S \rangle$ is returned.

Algorithm 1: UH$(V, P, T, K, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t)$.

1. $T_p \leftarrow \text{WW}(V, P, T, q_{ip}, t_i, c_{pt}, e_{pt})$
2. $S \leftarrow \emptyset$
3. **for** $t = 1$ **to** $|T|$ **do**
4. $\quad V_t \leftarrow \{i \in V : t_i = t\}$
5. $\quad B_t \leftarrow \text{BP}(V_t, Q, \bar{q}_i)$
6. $\quad$ **for each** $b \in B_t$ **do**
7. $\quad\quad S \leftarrow S \cup \langle b, t \rangle$
8. **return** $\langle T_p, S \rangle$

4.2. Lot Sizing and then Bin Packing heuristic (LSBP)

The LSBP heuristic initially solves the lot sizing problem. Then, it solves a number of bin packing subproblems to decide how to pack and when to deliver the orders. Note that if due to lot sizing decisions the quantities of an order i are all available before t_i, such order can be packed with orders having earlier due dates than i to reduce unused space and possibly decrease the number of bins. So, differently from UH, the LSBP heuristic tries to better exploit the structure of the packing subproblem by grouping orders having different due dates. Another improvement over UH is that LSBP also tries to identify if there is a range of periods a bin can be delivered at a lower cost.

The pseudo-code of LSBP is described in Algorithm 2. As in UH, the lot sizing subproblem is solved in line 1, and the set S is initialized in line 2. The for loop in lines 3–9 runs for each period $t \in T$. Let $t'_i \in \{1, \ldots, t_i\}$ be the earliest possible delivering date for order $i \in V$, *i.e.*, the earliest period when all quantities of order i are available according to the lot sizing solution. For each period $t \in T$, the subset $V_t \subseteq V$ of unpacked orders i whose earliest possible delivering date is $t'_i \leqslant t$ is identified in line 4. Note that in the LSBP heuristic the set of orders V_t composing the bin packing instance in each period t is not likely to coincide with the orders having $t_i = t$. The resulting bin packing solution is stored in $B_t \subset 2^{V_t}$ in line 5. Then, in the for loop in lines 6–9, LSBP evaluates if each bin b of B_t shall actually be delivered in period t or if it is advantageous to unpack the orders in b to be repacked and delivered in later periods. Let u_b be the delivery date of bin b, *i.e.*, the minimum due date among the orders packed in b. A bin b is delivered in t if $u_b = t$ or if, for $u_b > t$, t has the lowest delivering cost in the set $\{t, t+1, \ldots, u_b\}$. In this case, the tuple $\langle b, t \rangle$ is added to S in line 9. Otherwise, the bin is discarded and the orders in b are considered unpacked to the next period. LSBP stops in line 10, where the solution $\langle T_p, S \rangle$ is returned.

Algorithm 2: LSPB$(V, P, T, K, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t)$.

01. $T_p \leftarrow \text{WW}(V, P, T, q_{ip}, t_i, c_{pt}, e_{pt})$
02. $S \leftarrow \emptyset$
03. **for** $t = 1$ **to** $|T|$ **do**
04. $\quad V_t \leftarrow \{i \in V : t'_i \leqslant t \wedge \nexists \langle b, t_b \rangle \in S : i \in b\}$
05. $\quad B_t \leftarrow \text{BP}(V_t, Q, \bar{q}_i)$
06. $\quad$ **for each** $b \in B_t$ **do**
07. $\quad\quad u_b \leftarrow \min\{t_i : i \in b\}$
08. $\quad\quad$ **if** $t = u_b$ **or** $f_t < f_{\bar{t}}$, $\bar{t} \in \{t+1, \ldots, u_b\}$, **then**
09. $\quad\quad\quad S \leftarrow S \cup \langle b, t \rangle$
10. **return** $\langle T_p, S \rangle$

4.3. Combinatorial Relaxation-based Heuristic (CRH)

The CRH heuristic initially uses the $\text{CR}^{w_t \geqslant 0}$ formulation to decide (i) the periods in which there are purchases of each product in P and (ii) the delivery date of each order in V. Then, it solves a bin packing subproblem for each period in T to decide how to pack the orders that are delivered in the same period. The rationale behind this approach is that $\text{CR}^{w_t \geqslant 0}$ can be efficiently solved to optimality, giving a good partial solution with the purchase dates of products and the delivery dates of orders. Besides, the remaining decision of how to pack the items in each order can also be efficiently computed as only one bin packing subproblem problem needs to be solved, for each period in T, with only the orders that are delivered in that specific period.

The pseudo-code of CRH is described in Algorithm 3. In line 1, the $\text{CR}^{w_t \geqslant 0}$ formulation is solved. The periods in which there are purchases of product $p \in P$ are stored in $T_p \subseteq T$, and the orders that are delivered at period t are stored in $V_t \subseteq V$, for all $t \in T$. The set S is initialized in line 2. The for loop in lines 3–6 runs for each period $t \in T$, and a bin packing instance is solved whenever there are orders to be delivered at a period t. The resulting bin packing solution is stored in $B_t \subset 2^{V_t}$ in line 4. The for loop in lines 5 and 6 inserts each bin $b \in B_t$ in the solution S. CRH stops in line 7, where the solution $\langle T_p, S \rangle$ is returned.

Algorithm 3: CRH$(V, P, T, K, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t \rangle)$.

1. $\langle T_p, V_t \rangle \leftarrow \text{Solve-CR}^{w_t \geqslant 0}(V, P, T, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t)$
2. $S \leftarrow \emptyset$
3. **for** $t = 1$ **to** $|T|$ **such that** $V_t \neq \emptyset$ **do**
4. $\quad B_t \leftarrow \text{BP}(V_t, Q, \bar{q}_i)$
5. $\quad$ **for each** $b \in B_t$ **do**
6. $\quad\quad S \leftarrow S \cup \langle b, t \rangle$
7. **return** $\langle T_p, S \rangle$

4.4. Bin Packing and then Lot Sizing heuristic (BPLS)

The BPLS heuristic is based on the idea of first deciding how to pack the items and only then deciding when to buy the lots and when to deliver the orders. To this end, it initially generates the set Δ with all the partitions of T into intervals of time periods. For example, for $T = \{1, 2, 3, 4\}$, we have $\Delta = \{\delta^4\} \cup \{\delta^{3a}\} \cup \{\delta^{3b}\} \cup \{\delta^{3c}\} \cup \{\delta^{2a}\} \cup \{\delta^{2b}\} \cup \{\delta^{2c}\} \cup \{\delta^1\}$, with

$$\delta^4 = \{[1,1], [2,2], [3,3], [4,4]\},$$
$$\delta^{3a} = \{[1,1], [2,2], [3,4]\},$$
$$\delta^{3b} = \{[1,1], [2,3], [4,4]\},$$
$$\delta^{3c} = \{[1,2], [3,3], [4,4]\},$$

$$\delta^{2a} = \{[1,2],[3,4]\},$$
$$\delta^{2b} = \{[1,1],[2,4]\},$$
$$\delta^{2c} = \{[1,3],[4,4]\},$$
$$\delta^{1} = \{[1,4]\},$$

where $[l,u]$ denotes a subset of consecutive time periods starting with $l \in T$ and ending with $u \in T$. For each partition $\delta \in \Delta$, BPLS builds a solution as follows. First, it solves a bin packing instance for each subset in δ with $V_{[l,u]} = \{i \in V : l \leqslant t_i \leqslant u\}$. Then, it solves an instance of the lot sizing subproblem where the demands of each product $p \in P$ in each period $t \in T$ are set according to how the orders were packed into bins in the preceding step. Finally, after the lot sizing subproblem has been solved, BPLS delivers each bin at the period, among those it can be feasibly delivered, with the smallest cost.

As the size of Δ grows exponentially with $|T|$, the partitions in Δ are limited to those whose intervals have at most β time periods. For example, for $\beta = 3$, we have $\Delta = \{\delta^4\} \cup \{\delta^{3a}\} \cup \{\delta^{3b}\} \cup \{\delta^{3c}\} \cup \{\delta^{2a}\} \cup \{\delta^{2b}\} \cup \{\delta^{2c}\}$. We note that partition $\delta^1 = \{[1,4]\}$ is no longer in Δ because the number of time periods in $[1,4]$ is larger than β. Similarly, for $\beta = 2$, we have $\Delta = \{\delta^4\} \cup \{\delta^{3a}\} \cup \{\delta^{3b}\} \cup \{\delta^{3c}\}$. Furthermore, the rationale to limit the number of consecutive periods to β is not only to reduce the size of the bin packing subproblem, but also that, due to the holding costs, it is unlikely to have attractive solutions where orders from distant periods of the planning horizon are packed in the same bin.

The pseudo-code of BPLS is described in Algorithm 4. The for loop in lines 1–10 builds a solution $\langle T_p, S\rangle$ for each partition $\delta \in \Delta$. A bin packing instance is solved for each $[l,u] \in \delta$, and in line 2 the union of all the resulting bin packing solutions is stored in $B \subset 2^V$. The latest period $u_b \in T$ in which the bin $b \in B$ can be delivered is computed in line 3. As all the orders in b must be delivered at the same period, a new due date $\bar{t}_i = u_b$ is set for each order $i \in b$ in line 4. A lot sizing instance characterized by the tuple $\langle V, P, T, q_{ip}, \bar{t}_i, c_{pt}, e_{pt}\rangle$ is solved in line 5. Note that the periods in which the demands of each product $p \in P$ occurs in the lot sizing subproblem are possibly anticipated since we may have $\bar{t}_i < t_i$ for some orders due to packing decisions. In this case the cost of holding q_{ip} unities of p during periods $\bar{t}_i, \bar{t}_i + 1, \ldots, t_i - 1$ are computed before solving the lot sizing subproblem. The set S is initialized in line 6. The for loop in lines 7–9 decides the delivery date of each bin $b \in B$. Let l_b be the earliest possible delivery date for the bin $b \in B$, *i.e.*, the earliest period where all the quantities of orders $i \in b$ are available according to the lot sizing solution T_p. The period t^* with the smallest delivering cost for bin b is identified in line 8. Then, the tuple $\langle b, t^*\rangle$ is added to S in line 9, indicating that b is delivered at t^*. The best-known solution $\langle T_p^*, S^*\rangle$ is updated in line 10, and returned in line 11.

Algorithm 4: BPLS$(V, P, T, K, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t, \beta)$.

01. **for each** $\delta \in \Delta$ **do**
02. $\quad B \leftarrow \bigcup_{[l,u]\in\delta} \mathrm{BP}(V_{[l,u]} = \{i \in V : l \leqslant t_i \leqslant u\}, Q, \bar{q}_i)$
03. $\quad u_b \leftarrow \min\{t_i : i \in b\},\ \forall b \in B$
04. $\quad \bar{t}_i \leftarrow u_b,\ \forall i \in b : b \in B$
05. $\quad T_p \leftarrow \mathrm{WW}(V, P, T, q_{ip}, \bar{t}_i, c_{pt}, e_{pt})$
06. $\quad S \leftarrow \emptyset$
07. $\quad$ **for each** $b \in B$ **do**
08. $\quad\quad t^* \leftarrow \arg\ \min_{l_b \leqslant t \leqslant u_b} f_t$
09. $\quad\quad S \leftarrow S \cup \{\langle b, t^*\rangle\}$
10. $\quad$ **update** $(\langle T_p, S\rangle, \langle T_p^*, S^*\rangle)$
11. **return** $\langle T_p^*, S^*\rangle$

5. Computational experiments

We report computational experiments conducted on a set of six groups of instances to cover a variety of situations. A total of 360 instances were used. The MILP formulations SF and MF and their corresponding linear relaxation, as well as the combinatorial relaxation $\mathrm{CR}^{w_t \geqslant 0}$, were implemented in the Optimization Programming Language (OPL), which is supported by the IBM ILOG CPLEX Optimization Studio (version 12.6). Our aim is to assess the optimality gaps we can get in reduced computational times using lower and upper bounds. The former with the SF and MF linear relaxation and $\mathrm{CR}^{w_t \geqslant 0}$, and the latter with the proposed heuristics. The heuristics UH, LSBP, CRH, and BPLS were implemented in C++ and compiled with GCC (version 4.8.4). The value of β was set to 4 in BPLS. The computational experiments were run on an Intel(R) Xeon(R) Quad-Core E5405 CPU with 2.00 GHz of clock speed and 16 GB of RAM memory, running the Linux Ubuntu operating system (version 14.04).

5.1. Testbed instances

An instance is characterized by the tuple $\langle V, P, T, K, Q, q_{ip}, t_i, c_{pt}, e_{pt}, f_t \rangle$. The testbed instances used in the computational experiments are generated from three parameters: $\eta = |V|$, $\rho = |P|$, $\mu = |T|$. The due date of each order $i \in V$ is set to $U[1; \mu]$, where $U[a; b]$ denotes a pseudorandom number generated with an uniform distribution in the range $[a; b]$, see [10]. The value of Q was fixed in 10 000, and the computation of the values q_{ip} (relative to Q), for each $i \in V$ and $p \in P$, was inspired by the well known *Triplet* instances for the bin packing problem [8]. These instances are generated such that their optimal solution has three items per bin and no unused space. Following the methodology proposed by Falkenauer [8], we first build an instance of the bin packing problem, whose optimal solution has $|K| = \eta/3$ bins, by splitting each bin into three pieces. The size of the first piece is set to $U[0.38 \cdot Q; 0.49 \cdot Q]$, while the size of the second is set to $U[0.25 \cdot Q; Q/2]$. The remaining space in the bin corresponds to the size of the third piece. Next, we randomly assign each piece to the weight $\bar{q}_i$ for each order $i \in V$. We have that $\bar{q}_i = \sum_{p \in P} q_{ip}$, and we compute the exactly value of q_{ip} as follows. First, we set $q_{ip} = 0$ for all $i \in V$ and $p \in P$. Then, while $\sum_{p \in P} q_{ip} \leqslant \bar{q}_i$, we iteratively assign a demand $q_{ip} = \min\{\bar{q}_i - \sum_{p \in P} q_{ip},\ U[0.2 \cdot \bar{q}_i; 0.5 \cdot \bar{q}_i]\}$ to a randomly chosen product $p \in P$. This approach allows scenarios where different orders have demands for different subsets of items.

We generated six groups of instances that differ from each other by how the production costs c_{pt} and e_{pt}, and the delivering cost f_t, are computed. In the first group of instances, the setup cost was set to $c_{pt} = U[12\,000; 13\,000]$, and the inventory holding cost was set to $e_{pt} = U[0.4; 0.6]$ for each unity of the product $p \in P$ at period $t \in T$. The delivering cost for each period $t \in T$ was set to $f_t = \theta / \log(|K|)$, where $\theta = U[11\,000; 14\,000]$. These values of c_{pt}, e_{pt}, and f_t were chosen so that the contribution in the objective function of the sum of the production costs is approximately the same as that of the delivering costs. This leads to harder instances, as both the Lot Sizing and the Bin Packing subproblem must be well solved to find optimal or near-optimal solutions.

The remaining five groups were generated as those in the first group. However, in the second group, the values of c_{pt} and f_t, for all $p \in P$ and all even $t \in T$, were increased by 30%. In addition to this 30% increase in c_{pt} and f_t at the even periods, the instances in the third group had the value of c_{pt} and e_{pt} further increased by 20% at all periods in T. The instances in the fourth group had the value of c_{pt} and e_{pt} increased by 20%, while those in the fifth group had the value of the delivering cost increased by 20% at all periods. In the sixth group, the values of c_{pt} and f_t for all $p \in P$ and all even $t \in T$ were increased by 30% and the value of delivering cost was further increased by 20% at all periods.

Each of the six groups is divided into six sets of ten instances randomly generated with the following values of η, ρ and μ. The first and second sets have the same value of $\rho = 6$ and $\mu = 4$, but have $\eta = 36$ and $\eta = 48$, respectively, while the third and forth sets have $\rho = 6$ and $\mu = 6$, with $\eta = 72$ and $\eta = 90$, respectively. Besides, the fifth and sixth sets have the same value of $\rho = 8$ and $\mu = 8$, with $\eta = 120$ and $\eta = 144$, respectively. Each set of instances is identified by the number of the group, and the values of η, ρ and μ. For example, the name g1-36-6-4 refers to a set of instances from the first group that was generated with $\eta = 36$, $\rho = 6$ and $\mu = 4$.

Table 1 summarizes how the parameters in the six groups of instances are generated. The first line, gives the six different combinations of η, ρ and μ. Each combination represents a set of 10 instances for each of the six groups. Next, the following four lines show how to compute the values of Q, t_i, e_{pt}, and c_{pt}, respectively. Then, the final line displays how to obtain the value of θ that is used to compute the value of f_t, *i.e.* $f_t = \theta/\log(|K|)$. The resulting 360 instances are available online at the link https://ufsj.edu.br/prof_ngoulart/instances_for_iulsbp.php.

5.2. Results

We first evaluate the performance of our approaches to obtain lower bounds for IULSBP. We evaluate the quality of the lower bounds and the computational effort to obtain them. Table 2 shows average results of 10 instances per line, which are identified in the first column as follows: the group – the number of orders – the number of products – the number of periods (for instance, g1-36-6-4 means group 1 with $|V| = 36$, $|P| = 6$, and $|T| = 4$). Then, results for the SF and MF linear relaxations and for the combinatorial relaxation are presented in the subcolumns identified with SF_{LR}, MF_{LR}, and $\text{CR}^{w_t \geqslant 0}$, respectively. We report, the average relative optimality gap ($[ub^* - lb]/ub^*$) in percentage between the lower bound lb obtained by each approach and the cost ub^* of the best-known solution for each instance (found by any of the heuristics developed in this paper). At the end of each group, we also show the average relative optimality gap over all instances in the group. Besides, we give the average running times in seconds over the 10 instances in the set. It can be observed that the average relative optimality gaps of the lower bounds provided by SF were up to 31.9% (on the instances in set g4-144-8-8), while that of MF were up to 7.7% (on the instances in set g1-120-8-8). Besides, the maximum running time of the latter (41.1 s) was less than half of the former (90.7 s), even though the former has a larger number of variables and constraints. This can be explained by the fact that the CPLEX pre-processing algorithm is more efficient to reduce the size of MF than that of SF. The best lower bounds were obtained by the combinatorial relaxation $\text{CR}^{w_t \geqslant 0}$. Its average optimality gaps were never larger than 3.3%, and its average running time was never larger than 0.5 s. Besides, the performance of $\text{CR}^{w_t \geqslant 0}$ is consistently good over all of the six groups of instances. The difference between the largest and the smallest average relative gap of $\text{CR}^{w_t \geqslant 0}$ is $0.6\%(2.8\% - 2.2\%)$, while that of SF and MF was $8.0\%(24.1\% - 16.1\%)$ and $1.9\%(6.2\% - 4.3\%)$, respectively.

We then evaluate the performance of the heuristics UH, LSBP, CRH, and BPLS. We aim at assessing the trade-off between the quality of the upper bounds provided by each heuristic and the corresponding running times. In Table 3, we report, for each heuristic, the average relative optimality gap ($[ub - lb^*]/ub$) in percentage between the upper bound ub obtained by the heuristic and the value lb^* of the best-known lower bound for each instance (provided by $\text{CR}^{w_t \geqslant 0}$), as well as its the average running times. At the end of each group, we also show the average relative optimality gap over all instances in the group. It can be observed that with a straightforward heuristic, as UH, it is not possible to obtain good quality solutions, as the average optimality gaps of UH were up to 15.6% (on the instances in the set g6-36-6-4). Much better results were obtained with the heuristics that attempt to take into account the solution of one subproblem as input to the other. The average optimality gaps of LSBP were never larger than 6.7% (on the instances in the set g5-36-6-4), while that of CRH were never larger than 5.5% (on the instances in the set g5-90-6-6). The smallest average optimality gaps over all sets of instances were obtained with BPLS. Although the running times of CRH were smaller than those of BPLS, the average optimality gaps of the former were never larger than 3.3% and its average running times were never larger than one minute. Besides, the performance of BPLS is consistently good over all of the six groups of instances. The difference between the largest and the smallest average relative gap of BPLS is $0.4\%(2.7\% - 2.3\%)$, while that of UH, LSBP and CRH were $7.3\%(13.1\% - 5.8\%)$, $1.1\%(4.8\% - 3.7\%)$ and $2.2\%(4.8\% - 2.6\%)$ respectively.

6. Concluding remarks

We proposed mixed integer linear programming formulations, a combinatorial relaxation, and heuristics for the integrated uncapacitated lot sizing and bin packing problem. Computational experiments on instances with

TABLE 1. Summary on how to generate the six groups of instances, where $\eta = |V|$, $\rho = |P|$, $\mu = |T|$, and $U[a; b]$ denotes a pseudorandom number generated with an uniform distribution in the range $[a; b]$. The value of θ is used to compute f_t, *i.e.* $f_t = \theta / \log(|K|)$, where $|K| = \eta/3$.

	Instance groups					
Data	g1	g2	g3	g4	g5	g6
η-ρ-μ	36-6-4, 48-6-4, 72-6-6, 90-6-6, 120-8-8, and 144-8-8					
Q	10 000					
t_i	$U[1; \lvert T \rvert]$					
e_{pt}	$U[0.4; 0.6]$	$U[0.4; 0.6]$	$U[0.48; 0.72]$	$U[0.48; 0.72]$	$U[0.4; 0.6]$	$U[0.4; 0.6]$
c_{pt}	$U[12\,000; 13\,000]$	$U[12\,000; 13\,000]$ for odd t $U[15\,600; 16\,900]$ for even t	$U[14\,400; 15\,600]$ for odd t $U[18\,720; 20\,280]$ for even t	$U[14\,400; 15\,600]$	$U[12\,000; 13\,000]$	$U[12\,000; 13\,000]$ for odd t $U[15\,600; 16\,900]$ for even t
θ (f_t)	$U[11\,000; 14\,000]$	$U[11\,000; 14\,000]$ for odd t $U[14\,300; 18\,200]$ for even t	$U[11\,000; 14\,000]$ for odd t $U[14\,300; 18\,200]$ for even t	$U[11\,000; 14\,000]$	$U[13\,200; 16\,800]$	$U[13\,200; 16\,800]$ for odd t $U[17\,160; 21\,840]$ for even t

TABLE 2. Performance of the approaches to obtain lower bounds for IULSBP. The first two approaches consist in solving the linear relaxation of SF and MF, respectively, while the third consists in solving the combinatorial relaxation $\mathrm{CR}^{w_t \geqslant 0}$.

	$\mathrm{SF_{LR}}$		$\mathrm{MF_{LR}}$		$\mathrm{CR}^{w_t \geqslant 0}$	
Name	gap(%)	t(s)	gap(%)	t(s)	gap(%)	t(s)
g1-36-6-4	10.7	0.2	2.8	0.1	1.0	0.1
g1-48-6-4	15.4	0.6	5.8	1.0	2.5	0.1
g1-72-6-6	21.8	4.1	6.5	7.2	2.8	0.2
g1-90-6-6	24.1	9.3	6.4	10.6	3.0	0.2
g1-120-8-8	29.2	39.7	7.7	34.5	3.0	0.4
g1-144-8-8	29.3	59.9	6.8	41.1	2.6	0.4
Average:	21.8		6.0		2.5	
g2-36-6-4	8.8	0.2	2.2	0.3	1.5	0.1
g2-48-6-4	11.9	0.9	4.1	1.0	2.6	0.1
g2-72-6-6	18.0	5.5	5.9	3.9	3.1	0.1
g2-90-6-6	19.3	13.1	4.8	7.2	2.9	0.1
g2-120-8-8	23.8	55.4	5.7	19.1	2.3	0.2
g2-144-8-8	24.2	90.7	5.1	33.0	2.1	0.3
Average:	17.7		4.6		2.4	
g3-36-6-4	9.3	0.2	2.9	0.4	0.9	0.1
g3-48-6-4	12.6	0.9	5.0	0.9	2.5	0.1
g3-72-6-6	19.9	5.5	5.6	4.4	2.6	0.1
g3-90-6-6	19.8	11.8	4.9	6.9	2.5	0.1
g3-120-8-8	26.4	57.5	7.1	17.9	2.5	0.3
g3-144-8-8	26.4	85.9	5.7	32.0	2.0	0.3
Average:	19.1		5.2		2.2	
g4-36-6-4	12.7	0.2	3.7	0.3	1.8	0.1
g4-48-6-4	18.6	0.9	6.1	1.1	3.1	0.1
g4-72-6-6	24.6	4.4	6.4	4.8	2.8	0.1
g4-90-6-6	25.6	8.9	6.6	7.5	3.1	0.2
g4-120-8-8	31.1	42.8	7.2	22.5	2.3	0.4
g4-144-8-8	31.9	61.7	7.0	36.8	2.5	0.4
Average:	24.1		6.2		2.6	
g5-36-6-4	13.8	0.4	3.6	0.4	1.3	0.1
g5-48-6-4	13.5	1.0	4.8	0.9	3.0	0.1
g5-72-6-6	19.3	4.1	6.1	4.7	3.3	0.1
g5-90-6-6	21.3	9.1	5.8	7.5	3.0	0.1
g5-120-8-8	27.7	38.9	7.2	20.8	3.0	0.4
g5-144-8-8	30.5	64.8	6.6	34.7	2.9	0.5
Average:	21.0		5.7		2.8	
g6-36-6-4	9.7	0.3	2.7	0.4	1.3	0.1
g6-48-6-4	11.1	0.9	3.4	0.9	2.6	0.1
g6-72-6-6	15.9	5.4	4.7	4.4	2.7	0.1
g6-90-6-6	16.9	12.9	4.0	7.8	3.0	0.1
g6-120-8-8	21.9	54.1	5.9	17.5	2.7	0.2
g6-144-8-8	21.3	88.1	5.3	30.7	2.6	0.3
Average:	16.1		4.3		2.5	

TABLE 3. Performance of the heuristics UH, LSBP, CRH, and BPLS.

	UH		LSBP		CRH		BPLS	
Name	gap(%)	t(s)	gap(%)	t(s)	gap(%)	t(s)	gap(%)	t(s)
g1-36-6-4	9.1	0.6	4.5	4.5	3.9	0.2	1.0	1.1
g1-48-6-4	7.1	0.8	5.4	0.8	4.8	0.3	2.5	1.3
g1-72-6-6	6.7	2.9	4.4	9.9	4.7	0.5	2.8	7.6
g1-90-6-6	6.0	4.4	4.5	11.3	5.0	0.6	3.0	8.9
g1-120-8-8	5.5	10.9	4.1	12.7	4.5	0.9	3.1	47.9
g1-144-8-8	4.2	14.2	3.6	27.0	3.7	0.9	2.6	58.6
Average:	6.4		4.4		4.4		2.5	
g2-36-6-4	13.9	0.4	3.4	0.3	3.2	0.2	1.6	1.1
g2-48-6-4	13.5	2.6	3.1	2.0	3.4	0.2	2.6	1.3
g2-72-6-6	13.0	16.6	5.0	15.2	3.8	0.4	3.3	7.6
g2-90-6-6	12.3	13.5	5.2	23.7	2.9	0.4	2.9	9.0
g2-120-8-8	10.9	14.9	3.6	32.4	2.5	0.6	2.5	48.6
g2-144-8-8	10.1	62.9	4.4	52.2	2.2	0.8	2.1	57.8
Average:	12.3		4.1		3.0		2.5	
g3-36-6-4	13.0	0.4	4.4	0.4	2.2	0.2	1.3	1.1
g3-48-6-4	11.2	0.4	3.1	1.0	2.8	0.2	2.5	1.2
g3-72-6-6	10.7	2.4	4.3	14.8	2.9	0.4	2.6	7.6
g3-90-6-6	10.8	4.4	3.8	15.6	2.8	0.4	2.5	9.1
g3-120-8-8	9.6	5.7	4.2	30.5	2.7	0.8	2.5	49.1
g3-144-8-8	9.2	17.7	4.0	34.1	2.1	0.8	2.1	58.2
Average:	10.8		4.0		2.6		2.3	
g4-36-6-4	8.4	0.4	4.2	0.2	3.2	0.2	1.8	1.3
g4-48-6-4	6.6	2.4	3.7	7.8	4.3	0.3	3.1	1.3
g4-72-6-6	6.1	1.0	4.1	10.3	4.6	0.6	2.8	8.0
g4-90-6-6	5.1	3.0	3.9	16.8	4.8	0.6	3.3	9.2
g4-120-8-8	4.6	4.1	3.2	15.5	3.5	0.9	2.4	48.9
g4-144-8-8	4.2	28.2	3.3	26.5	3.8	1.0	2.5	60.2
Average:	5.8		3.7		4.0		2.7	
g5-36-6-4	9.0	0.1	6.7	9.0	5.0	0.3	1.3	1.1
g5-48-6-4	7.5	0.2	4.2	0.2	4.8	0.3	2.1	1.3
g5-72-6-6	7.6	0.3	4.6	1.1	4.9	0.5	3.3	7.5
g5-90-6-6	7.2	12.3	5.1	15.2	5.5	0.6	3.0	8.9
g5-120-8-8	5.7	14.1	4.5	25.1	4.3	0.8	3.0	48.1
g5-144-8-8	5.1	31.5	3.7	9.2	4.3	0.9	3.0	58.5
Average:	7.0		4.8		4.8		2.6	
g6-36-6-4	15.6	0.1	4.1	6.1	3.4	0.2	1.9	1.1
g6-48-6-4	14.4	0.1	3.8	14.2	3.4	0.2	2.6	1.3
g6-72-6-6	13.5	0.2	5.1	6.6	3.6	0.4	2.7	7.5
g6-90-6-6	13.1	1.9	5.0	18.0	3.1	0.4	3.0	8.9
g6-120-8-8	11.3	8.8	4.9	10.0	3.0	0.7	2.7	47.8
g6-144-8-8	10.8	4.5	4.0	29.6	2.6	0.7	2.6	59.2
Average:	13.1		4.5		3.2		2.6	

different configurations have shown that the best lower bounds were obtained by the $\mathrm{CR}^{w_t \geqslant 0}$ combinatorial relaxation, which provided lower bounds within 3.3% of the optimal solution, in less than one second, on average. The best upper bounds were obtained by the BPLS heuristic, whose solutions have a maximum optimality gap of 3.3% and a maximum running time smaller than 1 min, on average. Besides, The performance of BPLS was consistently good on all six groups of instances with a maximum average optimality gap of 2.7% on group g4.

BPLS, which solves the bin packing subproblem first and only then solves a lot-sizing subproblem (that is dependent on the first one), found better results than the other heuristics even on the instance group g4, where the lot-sizing costs (setup and inventory holding costs) are relatively higher than on the other groups of instances. In this case, the setup and holding costs have a greater influence on the value of the objective function. Therefore, algorithms that focus on minimizing the lot-sizing components of the objective function (such as LSBP) are expected to find better solutions than other algorithms that focus on minimizing the delivering cost (such as BPLS). The better results of BPLS over LSPB on this group can be explained by the fact that BPLS packs the items together in the first step, but the decision of when the packs are delivered and when the corresponding products are bought is optimally solved by the WW algorithm in its second step.

Future works may explore different strategies to formulate the assignment-based bin packing constraints in (2.7), such as the pattern-based formulation of [18] and the arc-flow formulation of [5]. Alternatively, other heuristic methods based on Metaheuristics can also be devised for the problem. Another interesting line of research is to consider the capacitated lot sizing variants of the problem. Although the formulations could be adapted to the capacitated case, regarding the heuristic development, capacities would force different choices of building strategies and would have an impact on the algorithms availability to obtain high quality feasible solutions.

Acknowledgements. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES), the Brazilian National Council for Scientific and Technological Development (CNPq), and the Foundation for Support of Research of the State of Minas Gerais, Brazil (FAPEMIG). Ravetti acknowledges partial support from Fundep.

References

[1] Y. Adulyasak, J. Cordeau and R. Jans, The production routing problem: a review of formulations and solution algorithms. *Comput. Oper. Res.* **55** (2015) 141–152.

[2] A. Azadeh, S. Elahi, M.H. Farahani and B. Nasirian, A genetic algorithm-taguchi based approach to inventory routing problem of a single perishable product with transshipment. *Comput. Ind. Eng.* **104** (2017) 124–133.

[3] J.F. Bard and N. Nananukul, The integrated production-inventory-distribution-routing problem. *J. Scheduling* **12** (2009) 257–280.

[4] N. Ben-Khedher and C.A. Yano, The multi-item joint replenishment problem with transportation and container effects. *Transp. Sci.* **28** (1994) 37–54.

[5] F. Brandão and J.P. Pedroso, Bin packing and related problems: General arc-flow formulation with graph compression. *Comput. Oper. Res.* **69** (2016) 56–67.

[6] L.C. Coelho, J. Cordeau and G. Laporte, Thirty years of inventory routing. *Transp. Sci.* **48** (2014) 1–19.

[7] K. Copil, M. Wörbelauer, H. Meyr and H. Tempelmeier, Simultaneous lotsizing and scheduling problems: a classification and review of models. *OR Spectr.* **39** (2017) 1–64.

[8] E. Falkenauer, A hybrid grouping genetic algorithm for bin packing. *J. Heuristics* **2** (1996) 5–30.

[9] W.-S. Lee, J.-H. Hana and S.-H. Cho, A heuristic algorithm for a multi-product dynamic lot-sizing and shipping problem. *Int. J. Prod. Econ.* **98** (2005) 204–214.

[10] M. Matsumoto and T. Nishimura, Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul.* **8** (1998) 3–30.

[11] G.M. Melega, S.A. de Araujo and R. Jans, Classification and literature review of integrated lot-sizing and cutting stock problems. *Eur. J. Oper. Res.* **271** (2018) 1–19.

[12] F. Molina, R. Morabito and S.A. de Araujo, MIP models for production lot sizing problems with distribution costs and cargo arrangement. *J. Oper. Res. Soc.* **67** (2016) 1395–1407.

[13] L.V. Norden and S.L.V. d. Velde, Multi-product lot-sizing with a transportation capacity reservation contract. *Eur. J. Oper. Res.* **165** (2005) 127–138.

[14] Y. Pochet, Mathematical programming models and formulations for deterministic production planning problems. Computational Combinatorial Optimization, edited by M. Junger and D. Naddef. In: Vol. 2241 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg (2001) 57–111.

[15] E. Sancak and F.S. Salman, Multi-item dynamic lot-sizing with delayed transportation policy. *Int. J. Prod. Econ.* **131** (2011) 595–603.

[16] V. Schmid, K.F. Doerner and G. Laporte, Rich routing problems arising in supply chain management. *Eur. J. Oper. Res.* **224** (2013) 435–448.

[17] K.E. Stecke and X. Zhao, Production and transportation integration for a make-to-order manufacturing company with a commit-to-delivery business mode. *Manuf. Serv. Oper. Manage.* **9** (2007) 123–224.

[18] P.H. Vance, C. Barnhart, E.L. Johnson and G.L. Nemhauser, Solving binary cutting stock problems by column generation and branch-and-bound. *Comput. Optim. App.* **3** (1994) 111–130.

[19] H.M. Wagner and T.M. Whitin, Dynamic version of the economic lot size model. *Manage. Sci.* **5** (1958) 89–96.