Les Cahiers du GERAD ISSN: 0711–2440

A branch-price-and-cut algorithm for a production-routing problem with short-lifespan products

I. Dayarian
G. Desaulniers

G–2016–41

June 2016

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A branch-price-and-cut algorithm for a production-routing problem with short-lifespan products

Iman Dayarian [a]

Guy Desaulniers [b]

H. Milton Stewart School of Industrial & Systems Engineering Georgia Institute of Technology, Atlanta (Georgia) USA [a]

GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Québec) Canada [b]

iman.dayarian@isye.gatech.edu
guy.desaulniers@gerad.ca

June 2016

Les Cahiers du GERAD

G–2016–41

Abstract: We study a rich production-routing problem with time windows arising at a catering services company. The production part consists of assembling the meals to deliver. It considers release times to ensure freshness of the products to be delivered and is also restricted by due times incurred by the constructed routes. Production employee shifts together with a compatible production schedule must be determined. The routing part consists of building vehicle routes that can contain multiple trips and must satisfy customer time windows and vehicle capacity. Routing and production costs, including a guaranteed minimum paid time for the drivers and the production employees, are minimized under various constraints. To solve this complex problem, we propose an exact branch-price-and-cut algorithm. We introduce a new branching rule that imposes on one branch a lower bound on the production costs. Computational results obtained on instances derived from real-world datasets show the effectiveness of this branching rule. Overall, our algorithm is able to solve instances with up to 25 orders and 4 products in less than 3 hours of computational time.

Keywords: Catering services, short-lifespan products, production-routing problem, branch-price-and-cut, branching rule

Acknowledgments: We are grateful to the Natural Sciences and Engineering Research Council of Canada for their financial support. We would also like to thank the personnel of our anonymous industrial partner who proposed the problem to us and provided datasets.

1 Introduction

Manufacturing companies increasingly need to integrate production scheduling and transportation planning to optimize both processes jointly (Mula et al. 2010). Optimal decisions may not only rely on the efficiency of the individual processes at different locations, but one needs to take into account the behavior of linked decision systems (Frazzon 2009, Scholz-Reiter et al. 2011). This paper addresses a real-life production-delivery application arising in a catering services company. The considered company provides food delivery and also catering services to different customers in Montreal. More precisely the service includes delivering meal boxes, containing an appetizer, a main course and a dessert or platters of, e.g., sandwiches or vegetables to a series of customers, based on their daily demands. Each service to a customer, so called an *order*, is identified by the customer's address, a time interval during which the customer expects the service to be provided and finally the list of different requested products (specific meal boxes or platters) including their quantities. Moreover, a service time is associated with each delivery to a customer. The orders for a given day are received at most at 6:00 PM of the previous day. An operational production-delivery plan is then prepared for the following day.

The preparation of different products is done through a two-phase procedure in a central kitchen. In the first phase, the meal box and platter components (e.g., sandwiches, desserts, etc.) are produced during the night or early morning. If needed, these components are stored in refrigerators. In the second phase, the meal boxes and platters are assembled to fulfill the list of orders. Note that, in practice, these two phases may slightly overlap.

When delivering an order, the products it contains must be fresh. According to the catering services company that proposed this problem to us, the freshness of the individual components is ensured when they are produced on the same day and properly stored before being assembled. However, after assembling the components into a meal box or a platter, their freshness is only guaranteed for a certain duration. For example, the humidity of a salad can transfer to the bread of a sandwich after a certain time, degrading the quality of the sandwich. In consequence, the assembly phase of an order cannot start too in advance of its delivery time. The products are thus said to have short lifespans (or to be perishable). On the other hand, the assembly of an order must be completed before the departure time of the vehicle that will deliver it. These constraints link the order assembly phase with the design of the delivery routes. This design process takes into account driver and vehicle availability. We assume that the drivers and the vehicles are homogeneous.

Given that the component production phase is independent of the product assembly phase and the vehicle routing phase, we focus in this work on the assembly and routing processes, assuming that the component production is accomplished prior to the assembly phase. In the rest of this paper, we use both *production* and *assembly* interchangeably to refer to the assembly phase.

Given a list of orders, the production-routing problem that we consider, hereafter called the *caterer production-routing problem* (CPRP), consists of determining a production schedule, feasible employee work shifts to produce the orders, as well as feasible driver schedules and vehicle routes to deliver these orders in their respective time windows. Several constraints must be taken into account and the objective consists of minimizing the sum of the production employee salaries, production setup costs, the driver salaries and the vehicle operating costs. The salaries are proportional to the time worked with a minimum guaranteed paid time.

Currently, the CPRP is solved using a sequential process, in which delivery routes are planned first, followed by assembly scheduling. As a drawback, the obtained solutions may be locally optimal because the constraints linking production and routing are ignored in the routing step. In particular, delays often occur in production due to a poor distribution of the vehicle departure times, incurring non-optimal production due times. The main purpose of this paper is to develop an exact algorithm which is capable of generating an integrated least-cost production-delivery plan in such a way that the linking constraints are taken into account simultaneously.

Various production-routing problems have been studied in the literature as surveyed by Chen (2010) and Schmid et al. (2013). These problems differ by the characteristics of the production and the routing parts

of the problem. For the routing part, one can consider individual and immediate delivery, batch delivery to a single customer, batch delivery to multiple customers with direct shipping to each customer or with shipping routes visiting multiple customers, as well as fixed delivery departure times. For the production part, the literature concentrates on the machine configuration: single, parallel, flowshop, multiplant, etc. To our knowledge, no paper incorporates shift planning in the production part.

One important problem in this class is the so-called production-routing problem (for a survey, see Adulyasak et al. 2015). This problem combines the lot-sizing problem with the vehicle routing problem and consists of simultaneously optimizing production, inventory, and routing decisions. The benefits of coordinating these decisions was initially discussed by Chandra and Fisher (1994). The literature now includes papers addressing variants of this problem using metaheuristics (Boudia et al. 2007, Bard and Nananukul 2008, Boudia and Prins 2009, Armentano et al. 2011, Adulyasak et al. 2014b), heuristics based on branch-and-price (Bard and Nananukul 2009, 2010), and exact algorithms (Archetti et al. 2011, Adulyasak et al. 2014a). This problem substantially differs from the CPRP which involves a single period and no inventory.

Motivated by applications involving short-lifespan products such as chemicals, ready-mix concrete and catering food, several authors have studied problems where the product to be produced and distributed expires within a certain time frame after production, enforcing delivery before expiration time. Here, we focus on some papers published in the last decade.

Armstrong et al. (2008) consider an integrated production-delivery system in which a single machine is responsible to process the orders that must be delivered in a single shipment based on a prespecified sequence. Each order must be delivered within a given time window and its lifespan starts as soon as it is produced. The objective of the problem is to choose a subset of orders to serve such that the total demand of the chosen orders is maximized. The authors prove that their problem is NP-hard and develop a branch-and-bound algorithm as well as a heuristic procedure for solving it. Viergutz and Knust (2014) extend the work of Armstrong et al. (2008). They expand their model for handling delays of the production start as well as non-identical production and distribution sequences. They introduce two metaheuristics and improve the branch-and-bound algorithm of Armstrong et al. (2008).

Geismar et al. (2008) consider a variant of the production-distribution problem, involving a product with a short lifespan. Orders are delivered in batches and are produced in lots. Delivery to the customer locations must occur as soon as a lot is produced. The objective is to minimize the total operating time, including the time to produce and to deliver the products to a set of geographically dispersed customers. They consider the presence of a single vehicle and propose a metaheuristic for this strongly NP-hard problem.

Chen et al. (2009) address a production-distribution problem with perishable products where the customer demands are stochastic, the customers have time windows and the objective is to maximize the total expected profit of the supplier. They develop an integrated integer nonlinear programming model that is solved following the decomposition of the model into two parts: production scheduling followed by vehicle routing. The production scheduling subproblem remains nonlinear and is solved using the Nelder-Mead method. A local search heuristic is devised to solve the vehicle routing part.

Amorim et al. (2013) study the impact of allowing the possibility to split the production of an order over several machines (lot-sizing decisions) while dealing with perishable products in a production-routing problem. The problem involves multiple products subject to sequence-dependent changeovers, while the products must be delivered by one of the available vehicles. In order to investigate the impact of splitting the lots, the authors design a series of experiments varying different key parameters and compare the solutions between two models, one allowing order splitting and the other disallowing it.

Very recently, Gao et al. (2015) investigate a production-distribution problem where a single vehicle is available with a prefixed route. Several trips along this route, skipping customers as needed, can be scheduled according to the production schedule that determines the order batches. The objective consists of minimizing the total completion time. The authors study first the optimal solutions of two special cases and devise a heuristic with a guaranteed performance.

Finally, Belo-Filho et al. (2015) consider the same problem as Amorim et al. (2013) that incorporates production scheduling, line-assignment, lot-sizing/splitting decisions in a multiple perishable product environment. The authors propose an efficient adaptative large neighborhood search heuristic that relies on mixed-integer programming tools.

The contribution of this paper is three-fold. First, we introduce a complex production-routing problem, the CPRP, derived from a real-life application. To the best of our knowledge, several features of the CPRP (such as devising work shifts and taking into account guaranteed minimum paid time) have not been studied yet in the literature. Second, we develop an exact branch-price-and-cut algorithm for solving the CPRP. This algorithm includes an innovative branching rule that imposes a lower bound on the production cost on one branch and a delay of at least one order due time on the other. The lower bound computation relies on the solution of a mixed integer linear program (MILP) and our computational results show that the time spent solving these programs is compensated by a smaller search tree. We believe that this new branching rule can be adapted to other types of problems. Finally, we perform extensive computational experiments on instances derived from real-life datasets. Our computational results show the limit of the proposed solution method and allow to measure the gains that can be achieved by integrating production and routing in the catering services context.

The paper is organized as follows. In Section 2, we describe the CPRP in more detail and introduce some notation. Section 3 provides a mathematical formulation for this problem, while Section 4 describes the proposed solution algorithm. In Section 5, we investigate the performance of this algorithm through a series of computational tests and we estimate the value of using an integrated approach for solving the CPRP. Finally, Section 6 contains our concluding remarks.

2 Problem statement and notation

The CPRP consists of preparing the production-delivery plan for a given day of operations. This day is partitioned into a set of time periods of equal length (e.g., 15 or 30 minutes). Let $\mathcal{T}$ be this set of periods that are numbered chronologically from 1 to $|\mathcal{T}|$. The production employee shifts and the driver schedules must start and end at the beginning and the end of a period in $\mathcal{T}$, respectively. Each period is divided into disjoint subperiods of equal length (e.g., 5 or 10 minutes). In production, it is assumed that an employee is assigned to a maximum of one product during one subperiod and that the setup time required prior to the start of the production of a new product is an integer number of subperiods. The set of subperiods is denoted $\mathcal{U}$ and the subperiods are numbered from 1 to $|\mathcal{U}|$. The start and end times of period $t \in \mathcal{T}$ are denoted b^t and e^t, respectively, where $e^t = b^{t+1}$ for $t \in \mathcal{T} \setminus \{|\mathcal{T}|\}$. Similarly, we denote b^u and e^u, the start and end times of subperiod $u \in \mathcal{U}$.

Let $\mathcal{P}$ be the set of products. For each product $p \in \mathcal{P}$, we are given a production rate η^p per subperiod (not necessarily an integer) and a maximum allowed time (in minutes) between production and delivery μ^p, representing the product's lifespan (typically, 150 to 300 minutes). Let $\mathcal{O}$ be the set of orders (or customers). Each order $i \in \mathcal{O}$ is defined by a customer location, a demand for each product d_i^p, a space requirement ν_i, a service time τ_i^S, and a time window $[\underline{\omega}_i, \overline{\omega}_i]$ during which the delivery must be accomplished. We denote by $\mathcal{P}_i$ the set of products with a positive demand. Typically, the width of the time window is relatively small: in general, 15 or 30 minutes, but never more than 60 minutes. In consequence, we use the middle of this time window to approximate the delivery time and to compute a production release time for each ordered product in $\mathcal{P}_i$. More precisely, for each product $p \in \mathcal{P}_i$, this release time is set as the end of the subperiod containing time $0.5(\underline{\omega}_i + \overline{\omega}_i) - \mu^p$. We denote by $\mathcal{U}_i^p$ the set of subperiods during which the demand for product p in order i can be produced according to this release time and the minimum time for a driver to prepare his/her vehicle and reach customer i from the production facility. Let $\mathcal{U}^p = \bigcup_{i \in \mathcal{O}} \mathcal{U}_i^p$.

The production of the orders is performed at a single location (the production facility) by a maximum of n^E production employees on a maximum of n^W workstations (typically, $n^W < n^E$). One employee at a time may be assigned to a workstation. The employees (resp. workstations) are considered identical. Work shifts must be determined for the working employees. A shift starts at the beginning of a period and ends at the end of another one. Its length must not exceed a maximum duration $\bar{l}$ (a number of periods). For

each subperiod in a shift, the employee can be assigned to the production of a product (for one or several orders), to a setup or to an idle time. A setup that lasts σ minutes (a multiple of the subperiod length) must be assigned before starting the production of a new product. Figure 1 illustrates the solution of a CPRP. Three shifts requesting two workstations are displayed in its bottom part. In these shifts, the black rectangles correspond to the assignment to a product, whereas the grey ones represent the assignment to a different product. The employee is idle in the white rectangles. In the subperiods marked with S, the employee performs a setup.

The production of an order can be split on several workstations and preempted. Preemption is not ideal to avoid operational errors but it does not often occur in practice given the relatively small number of products and the relatively high quantity of units to produce. Once the items of an order are assembled, they are put together in one or several transportation containers and moved to a refrigerated storage room. Given the short lifespan of the products, the orders do not stay very long in the storage room and, thus, we assume that its capacity is not binding.

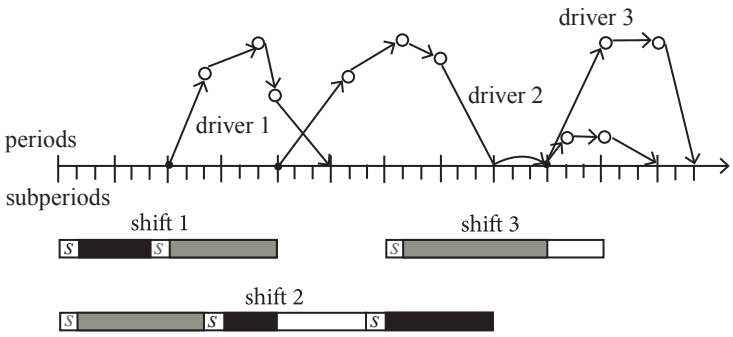

Figure 1: Example of a solution to the CPRP

The production costs are given by the sum of the employee salaries and the setup costs. The salaries are computed according to a rate of γ^E per subperiod but a minimum number of paid subperiods $\underline{l}$ per shift is guaranteed. A setup cost γ^S is incurred for each setup performed. The setup costs are secondary and only used to avoid switching unnecessarily from one product to another in the same shift.

Deliveries are performed through a series of routes, carried out by a homogeneous fleet of vehicles and drivers. There are n^V vehicles and n^D drivers available. Each vehicle has a spatial capacity of Q. A driver on duty is assigned to a route that may contain one or several trips with a maximum of m trips. A trip starts at the depot (production facility) at the beginning of a period and is composed of a constant preparation time τ^P (to check the orders and load the vehicle), a sequence of travels, possible waitings (for the opening of a time window), and deliveries, followed by a return travel to the depot. A trip finishes at the end of the return period (possibly with a short idle period for the driver). A vehicle is assigned to a driver for the complete span of his/her route even if it contains several trips and waiting between consecutive trips. A vehicle can be used by several drivers if their routes do not overlap. A route is feasible if each of its trip satisfies vehicle capacity and the visited customers' time windows. More precisely, for a trip delivering the k orders $i_1, i_2, \ldots, i_k$ in this order, the vehicle capacity is satisfied if $\sum_{j=1}^{k} \nu_{i_j} \leq Q$. Furthermore, if i_0 denotes the depot, H_{i_0} the start time of the trip, $\tau^S_{i_0} = \tau^P$ the trip preparation time, and h_{ij} the traveling time between locations i and j, then the start of service time H_{i_j} at each customer i_j is computed recursively as:

$$H_{i_j} = \max\{\underline{\omega}_{i_j}, H_{i_{j-1}} + \tau^S_{i_{j-1}} + h_{i_{j-1},i_j}\}, \qquad j = 1, 2, \ldots, k.$$

The customers' time windows are satisfied if $H_{i_j} \leq \overline{\omega}_{i_j}$ for all $j = 1, 2, \ldots, k$. Obviously, when the route contains several trips, the end time of a trip cannot be later than the start time of the following trip if any. Finally, the total duration of a route must not exceed a maximum duration $\overline{l}$ (the same as the maximum shift length). Note that the preparation, service, and travel times as well as the time windows do not have to be expressed as multiples of the subperiod length. Figure 1 illustrates in its top part the routes assigned to three drivers (the circles represent customers). Only two vehicles are needed for these routes because those

assigned to drivers 1 and 3 do not overlap. Observe that driver 2 is assigned to a two-trip route. For each order $i \in \mathcal{O}$, we denote by $\mathcal{T}_i \subseteq \mathcal{T}$ the subset of periods at the beginning of which a trip can start to deliver this order within its time window.

The routing costs are composed of the driver salaries and vehicle operating costs. The salaries are computed as for the production employees, i.e., using a rate γ^D per minute with a guaranteed minimum paid time $\underline{l}^D$ (the same used for the production employees but expressed in minutes). The vehicle operating costs are proportional to the total travel time at a rate of γ^V per minute.

Given that the production release times of the order products have been set according to the orders' time windows, the production phase is only linked to the routing phase by the following constraint. Each trip of a route cannot start its preparation unless all orders that must be delivered along this trip have been produced. Conversely, we can state this constraint as follows: all orders to be delivered in a trip of a route must be produced before the trip preparation starts. Consequently, the trip start times impose production due times for the orders.

To summarize, the CPRP consists of establishing feasible production employee shifts with a compatible feasible production schedule, as well as feasible delivery routes such that the total production and routing costs are minimized while satisfying various operational constraints, including product freshness upon delivery.

3 Mathematical model

In this section, we formulate the CPRP. Beforehand, we introduce additional notation.

Let $\mathcal{R}$ be the set of feasible delivery routes. For each route $r \in \mathcal{R}$, denote by c_r its cost which includes the driver salary and the vehicle operational cost. Furthermore, we define, for each customer $i \in \mathcal{O}$, a binary parameter a_{ir} that is equal to 1 if route r visits customer i and 0 otherwise. Also, for each customer $i \in \mathcal{O}$ and each period $t \in \mathcal{T}_i$, a binary parameter g^t_{ir} indicates whether or not customer i is visited in route r on a trip starting at time b^t. Finally, for each period $t \in \mathcal{T}$, the binary parameter b^t_r (resp. e^t_r) is equal to 1 if route r starts at time b^t (resp. ends at time e^t) and 0 otherwise.

Let $\mathcal{S}$ be the set of feasible production employee shifts, where a shift also describes the employee assignment in each of its subperiods. For each shift $s \in \mathcal{S}$, denote by c_s its cost which includes the employee salary and the setup costs. Also, for each product $p \in \mathcal{P}$ and each subperiod $u \in \mathcal{U}^p$, the binary parameter f^{pu}_s is equal to 1 if shift s is assigned to product p in subperiod u. For each period $t \in \mathcal{T}$, the binary parameter b^t_s (resp. e^t_s) indicates whether or not shift s starts at time b^t (resp. ends at time e^t).

The proposed model relies on five types of variables. For each route $r \in \mathcal{R}$, let θ_r be a binary variable equal to 1 if route r is selected and 0 otherwise. For each shift $s \in \mathcal{S}$, let ψ_s be an integer variable indicating the number of production employees working on shift s. Next, the nonnegative variable q^{pu}_i represents the number of units of product $p \in \mathcal{P}$ produced in subperiod $u \in \mathcal{U}$ for order $i \in \mathcal{O}$. Finally, for each period $t \in \mathcal{T}$, the integer variables v^t and w^t give the numbers of vehicles and workstations used, respectively.

Using this notation, the CPRP can be formulated as the following MILP:

$$\min \quad \sum_{r\in\mathcal{R}} c_r\theta_r + \sum_{s\in\mathcal{S}} c_s\psi_s \tag{1}$$

$$\text{s.t.} \quad \sum_{r\in\mathcal{R}} a_{ir}\theta_r = 1, \qquad \forall i \in \mathcal{O} \tag{2}$$

$$\sum_{i\in\mathcal{O}^p_u} q^{pu}_i \le \eta^p \sum_{s\in\mathcal{S}} f^{pu}_s\psi_s, \qquad \forall p \in \mathcal{P}, u \in \mathcal{U}^p \tag{3}$$

$$\sum_{\substack{u\in\mathcal{U}^p_i \\ e^u\le b^t}} q^{pu}_i \ge \sum_{r\in\mathcal{R}} d^p_i g^t_{ir}\theta_r, \qquad \forall i \in \mathcal{O}, p \in \mathcal{P}_i, t \in \mathcal{T}_i \tag{4}$$

$$\sum_{r\in\mathcal{R}} \theta_r \le n^D, \tag{5}$$

$$\sum_{s\in\mathcal{S}} \psi_s \leq n^E, \tag{6}$$

$$v^{t-1} + \sum_{r\in\mathcal{R}} (b_r^t - e_r^{t-1})\theta_r = v^t, \qquad \forall t \in \mathcal{T} \tag{7}$$

$$w^{t-1} + \sum_{s\in\mathcal{S}} (b_s^t - e_s^{t-1})\psi_s = w^t, \qquad \forall t \in \mathcal{T} \tag{8}$$

$$\theta_r \in \{0,1\}, \qquad \forall r \in \mathcal{R} \tag{9}$$

$$\psi_s \geq 0, \quad \textit{integer}, \qquad \forall s \in \mathcal{S} \tag{10}$$

$$q_i^{pu} \geq 0, \qquad \forall i \in \mathcal{O}, p \in \mathcal{P}_i, u \in \mathcal{U}_i^p \tag{11}$$

$$0 \leq v^t \leq n^V, \quad \textit{integer}, \qquad \forall t \in \mathcal{T} \tag{12}$$

$$0 \leq w^t \leq n^W, \quad \textit{integer}, \qquad \forall t \in \mathcal{T}. \tag{13}$$

Objective function (1) aims at minimizing the total routing and production costs. Constraint set (2) guarantees that each customer is visited exactly once. Constraints (3) limit the quantity of each product that can be produced in each subperiod according to the number of production employees assigned to this product. Constraints (4) ensure that the production of each order is completed before the due time incurred by its corresponding delivery route. Constraints (5) and (6) limit the numbers of routes and shifts to the numbers of drivers and production employees available. Vehicle availability is imposed through constraints (7) and the upper bounds in (12). Similarly, workstation availability is enforced through constraints (8) and the upper bounds in (13). In (7) and (8) for $t = 1$, we set $v^0 = w^0 = e_r^0 = e_s^0 = 0$ for all $r \in \mathcal{R}$ and $s \in \mathcal{S}$. Finally, constraints (9)–(13) restrict the feasible domains of the decision variables.

The solution algorithm described in the next section uses the following two constraint sets that are redundant with (4):

$$\sum_{i\in\mathcal{O}} \sum_{\substack{u\in\mathcal{U}_i^p \\ e^u \leq e^t}} q_i^{pu} \geq \sum_{i\in\mathcal{O}} \sum_{r\in\mathcal{R}} \sum_{\substack{j\in\mathcal{T} \\ j\leq t}} d_i^p g_{ir}^j \theta_r, \qquad \forall p \in \mathcal{P}, t \in \mathcal{T} \tag{14}$$

$$\sum_{u\in\mathcal{U}_i^p} q_i^{pu} \geq d_i^p, \qquad \forall i \in \mathcal{O}, p \in \mathcal{P}_i. \tag{15}$$

Constraints (14) correspond to an aggregated version of constraints (4) over the orders. They impose lower bounds (computed from the vehicle trips) on the number of units of each product that must be produced between the start of the day and the beginning of every period, regardless of the orders to which the units are dedicated. Constraints (15) ensure that a sufficient number of units of each product is produced for each order over its possible production periods.

4 A branch-price-and-cut algorithm

Constraints (4) constitute the largest set of constraints in model (1)–(13). To reduce their impact on the effectiveness of the solution process, we relax these constraints initially and introduce them only as needed. Constraints (14) and (15) are, however, added to the model to avoid generating too many relaxed constraints. Note that the number of constraints in these two sets is much less than in set (4). In this section, we describe a branch-price-and-cut algorithm for the CPRP assuming, for the sake of the presentation, that all constraints (4) have been generated.

In practice, the numbers of elements in $\mathcal{R}$ and $\mathcal{S}$ are so large that it is not possible to enumerate them all or to solve model (1)–(15) directly using a mixed integer programming solver. To alleviate this difficulty, we resort to a column generation algorithm embedded in a branch-and-cut framework to derive integer solutions. This yields a so-called branch-price-and-cut algorithm (see Barnhart et al. 1998, Lübbecke and Desrosiers 2005).

The components of our algorithm, namely, column generation, valid inequalities, and branching rules, are discussed in Sections 4.1, 4.2, and 4.3, respectively.

4.1 Column generation

Column generation is applied at each node of the search tree to solve the associated linear relaxation which is called the master problem. In this section, we consider the master problem at the root node, namely, the linear relaxation of (1)–(15). At each iteration of a column generation algorithm, a master problem restricted to a subset of the variables and one or several pricing subproblems are solved. This restricted master problem (RMP) yields a primal and a dual solution. The subproblems aim at finding negative reduced cost columns (variables) among those that are not yet considered in the RMP or at proving that no such columns exist. In the former case, columns are added to the RMP before starting a new iteration. In the latter one, the algorithm stops.

For model (1)–(15), the variables θ_r, $r \in \mathcal{R}$, and ψ_s, $s \in \mathcal{S}$, are generated by column generation as needed using two types of pricing subproblems. The description of these subproblems requires the dual variables of the master problem. Let $(\pi_i^2)_{i\in\mathcal{O}}$, $(\pi_{pu}^3)_{p\in\mathcal{P},u\in\mathcal{U}^p}$, $(\pi_{ipl}^4)_{i\in\mathcal{O},p\in\mathcal{P}_i,l\in\mathcal{T}_i}$, π^5, π^6, $(\pi_t^7)_{t\in\mathcal{T}}$, $(\pi_t^8)_{t\in\mathcal{T}}$, and $(\pi_{pl}^{14})_{p\in\mathcal{P},l\in\mathcal{T}}$ be the duals associated with constraint sets (2)–(8) and (14), respectively.

4.1.1 Route generation

The reduced cost $\bar{c}_r$ of a route variable θ_r, $r \in \mathcal{R}$, is given by

$$\bar{c}_r = c_r - \sum_{i\in\mathcal{O}} a_{ir}\pi_i^2 + \sum_{i\in\mathcal{O}}\sum_{p\in\mathcal{P}_i}\sum_{t\in\mathcal{T}_i} d_i^p g_{ir}^t \pi_{ipt}^4 - \pi^5 - \sum_{t\in\mathcal{T}}(b_r^t - e_r^{t-1})\pi_t^7 + \sum_{p\in\mathcal{P}}\sum_{t\in\mathcal{T}}\sum_{\substack{j\in T\\ j\le t}} d_i^p g_{ir}^j \pi_{pt}^{14}. \tag{16}$$

To generate negative reduced cost variables θ_r, $r \in \mathcal{R}$, we use $|\mathcal{T}|$ pricing subproblems, namely one per period in $\mathcal{T}$. The subproblem associated with period t can generate routes starting at the beginning of period t, i.e., at time b^t. It is defined on a network $\mathcal{G}_t^R = (\mathcal{V}_t^R, \mathcal{A}_t^R)$, where $\mathcal{V}_t^R$ and $\mathcal{A}_t^R$ are its node and arc sets, respectively (see Figure 2). The node set $\mathcal{V}_t^R$ contains: one origin node o representing the start of a route at time b^t; one destination node $\bar{o}$ representing the end of a route at maximum time $e^{\bar{t}}$ with $\bar{t} = \min\{t + \bar{l} - 1, |\mathcal{T}|\}$; for each $l \in \mathcal{T}$ such that $t \le l \le \bar{t}$, one depot node o_l^D representing time e^l at the depot; and for each order $i \in \mathcal{O}$ that can be serviced by a feasible route starting at time b^t, one customer node o_i^C. The subsets of the depot nodes and of the customer nodes are denoted $\mathcal{V}_t^{R,D}$ and $\mathcal{V}_t^{R,C}$, respectively, so that $\mathcal{V}_t^R = \{o, \bar{o}\} \cup \mathcal{V}_t^{R,D} \cup \mathcal{V}_t^{R,C}$. A time window $[\underline{\alpha}_j, \overline{\alpha}_j]$ is associated with each node $j \in \mathcal{V}_t^R$: it is equal to $[b^t, b^t]$ for node o, $[e^{\bar{t}}, e^{\bar{t}}]$ for node $\bar{o}$, $[e^l, e^l]$ for each node o_l^D, and $[\underline{\omega}_i, \overline{\omega}_i]$ for each node o_i^C.

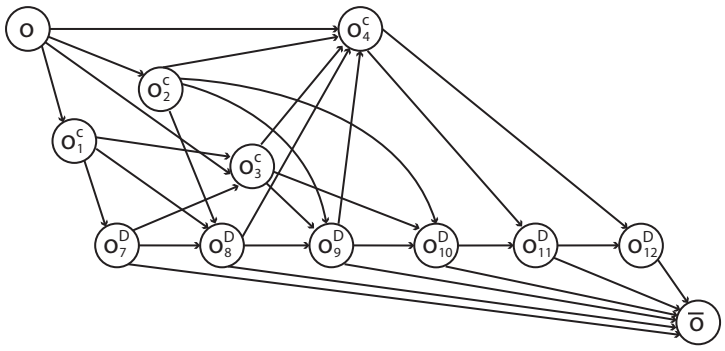

Figure 2: Example of a network $\mathcal{G}_t^R$ with four customer nodes

The arc set $\mathcal{A}_t^R$ comprises the following arcs. For each node $o_i^C \in \mathcal{V}_t^R$, there is an arc (o, o_i^C) representing the start of the first trip on a route that begins at time b^t and visits customer i first. For each node $o_l^D \in \mathcal{V}_t^R$, there is an arc $(o_l^D, \bar{o})$ representing the end of the last trip on a route in the interval $]b^l, e^l]$. For each node $o_l^D \in \mathcal{V}_t^R$ and each node $o_i^C \in \mathcal{V}_t^R$ that can be reached from the depot at the beginning of a trip starting at time b^{l+1} $(= e^l)$, there is an arc (o_l^D, o_i^C) representing the start of a trip (other than the first trip on a route), which begins at time b^{l+1} and visits customer i first. For each node $o_i^C \in \mathcal{V}_t^R$ and each node $o_l^D \in \mathcal{V}_t^R$ such that the depot can be reached in the interval $]b^l, e^l]$ directly after servicing customer i within its time

window, there is an arc (o_i^C, o_l^D) representing the end of a trip after visiting customer i. Finally, for each pair of nodes $o_{i_1}^C, o_{i_2}^C \in \mathcal{V}_t^R$ such that customer i_2 can be visited immediately after customer i_1, there is an arc $(o_{i_1}^C, o_{i_2}^C)$ representing a travel between these two customers. With each arc $(j,k) \in \mathcal{A}_t^R$, we associate a duration h_{jk}^+ and a partial cost c_{jk} (see Table 1). The durations are used to track the length of the routes and to ensure that the time windows are respected. The partial costs are needed to compute the reduced cost of the routes. They include the vehicle costs, the salary of the minimum time paid to the driver and some dual values. The computation of the complete reduced cost of the routes is performed while solving the subproblem on network $\mathcal{G}_t^R$.

Table 1: Duration and partial cost of the arcs in a network $\mathcal{G}_t^R$

arc (j,k)	duration h_{jk}^+	partial cost c_{jk}
(o, o_i^C)	$\tau^P + h_{jk}$	$\gamma^D \underline{l}^D + \gamma^V h_{jk} - \pi^5 - \pi_t^7$
$(o_l^D, \overline{o})$	0	π_{l+1}^7
(o_l^D, o_i^C)	$\tau^P + h_{jk}$	$\gamma^V h_{jk}$
(o_i^C, o_l^D)	$\tau_j^S + h_{jk}$	$\gamma^V h_{jk}$
$(o_{i_1}^C, o_{i_2}^C)$	$\tau_j^S + h_{jk}$	$\gamma^V h_{jk}$

Every feasible route in $\mathcal{R}$ corresponds to a path from o to $\overline{o}$ in $\mathcal{G}_t^R$. However, the opposite is not true as some $o - \overline{o}$ paths in $\mathcal{G}_t^R$ may not be feasible. Nevertheless, resource constraints can be used to enforce route feasibility. The pricing subproblem associated with routes starting at time b^t, $t \in \mathcal{T}$, corresponds to an elementary shortest path problem with resource constraints (Irnich and Desaulniers 2005) defined on network $\mathcal{G}_t^R$. It can, thus, be solved using a labeling algorithm. In this algorithm, a partial path from node o to a node $j \in \mathcal{V}_t^R$ is represented by a vector $\boldsymbol{\lambda} = (\lambda^{cost}, \lambda^{time}, \lambda^{load}, \lambda^{trip}, \lambda^{stp}, (\lambda^{cust_i})_{i\in\mathcal{O}})$ called a label: λ^{cost} is the path reduced cost; λ^{time} is the earliest ready time at node j; λ^{load} is the load accumulated in the current trip; λ^{trip} is the number of trips in the path; λ^{stp} is the start period of the current trip; and λ^{cust_i}, $i \in \mathcal{O}$, indicates whether or not customer i is unreachable from node j. A customer is said to be unreachable if it has been visited along the path or if it cannot be reached anymore before the end of its time window.

Starting from the initial label $(0, b^t, 0, 0, t, (0)_{i\in\mathcal{O}})$ associated with node o, a labeling algorithm extends the labels forwardly in network $\mathcal{G}_t^R$ to enumerate feasible partial paths, all starting from node o. The extension along an arc $(j,k) \in \mathcal{A}_t^R$ of a label $\boldsymbol{\lambda}_j = (\lambda_j^{cost}, \lambda_j^{time}, \lambda_j^{load}, \lambda_j^{trip}, \lambda_j^{stp}, (\lambda_j^{cust_i})_{i\in\mathcal{O}})$ associated with node j yields a new label $\boldsymbol{\lambda}_k = (\lambda_k^{cost}, \lambda_k^{time}, \lambda_k^{load}, \lambda_k^{trip}, \lambda_k^{stp}, (\lambda_k^{cust_i})_{i\in\mathcal{O}})$ whose components are computed using the following resource extension functions:

$$\lambda_k^{time} = \max\{\underline{\omega}_k, \lambda_j^{time} + h_{jk}^+\} \tag{17}$$

$$\lambda_k^{load} = \begin{cases} \lambda_j^{load} + \nu_k & \text{if } k \in \mathcal{V}_t^{R,C} \\ 0 & \text{if } k \in \mathcal{V}_t^{R,D} \\ \lambda_j^{load} & \text{otherwise} \end{cases} \tag{18}$$

$$\lambda_k^{trip} = \begin{cases} \lambda_j^{trip} + 1 & \text{if } j \notin \mathcal{V}_t^{R,C} \text{ and } k \in \mathcal{V}_t^{R,C} \\ \lambda_j^{trip} & \text{otherwise} \end{cases} \tag{19}$$

$$\lambda_k^{stp} = \begin{cases} l+1 & \text{if } j \in \mathcal{V}_t^{R,D} \text{ and } j = o_l^D \\ \lambda_j^{stp} & \text{otherwise} \end{cases} \tag{20}$$

$$\lambda_k^{cust_i} = \begin{cases} \lambda_j^{cust_i} + 1 & \text{if } k \in \mathcal{V}_t^{R,C} \text{ and } k = o_i^C \\ \max\{\lambda_j^{cust_i}, 1\} & \text{if } (k \notin \mathcal{V}_t^{R,C} \text{ or } k \neq o_i^C) \text{ and } \lambda_k^{time} + h_{k,o_i^C}^+ > \overline{\omega}_{o_i^C} \\ \lambda_j^{cust_i} & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{O} \tag{21}$$

$$\lambda_k^{cost} = \begin{cases} \lambda_j^{cost} + c_{jk} + \gamma^D \delta^R(\lambda_k^{time}, \lambda_j^{time}) + \sum_{p\in\mathcal{P}_i} d_i^p(\pi^4_{ip\lambda_k^{stp}} + \pi^{14}_{p\lambda_k^{stp}}) & \text{if } k \in \mathcal{V}_t^{R,C} \text{ and } k = o_i^C \\ \lambda_j^{cost} + c_{jk} + \gamma^D \delta^R(\lambda_k^{time}, \lambda_j^{time}) & \text{otherwise} \end{cases} \tag{22}$$

where

$$\delta^R(\lambda_k^{time}, \lambda_j^{time}) = \begin{cases} 0 & \text{if } \lambda_k^{time} \leq \underline{l}^D \\ \lambda_k^{time} - \max\{\lambda_j^{time}, \underline{l}^D\} & \text{otherwise} \end{cases}$$

gives the time exceeding the minimum paid time that must be paid along arc (j,k). Label $\boldsymbol{\lambda_k}$ is associated with node k and is deemed feasible only if $\lambda_k^{time} \leq \overline{\omega}_k$, $\lambda_k^{load} \leq Q$, $\lambda^{trip} \leq m$, and $\lambda_k^{cust_i} \leq 1$, $\forall i \in \mathcal{O}$. If one of these conditions is violated, label $\boldsymbol{\lambda_k}$ is discarded.

To avoid enumerating all feasible $o - \overline{o}$ paths, the following dominance rule is applied.

Definition 1 *Let $\boldsymbol{\lambda_\ell} = (\lambda_\ell^{cost}, \lambda_\ell^{time}, \lambda_\ell^{load}, \lambda_\ell^{trip}, \lambda_\ell^{stp}, (\lambda_\ell^{cust_i})_{i\in\mathcal{O}})$, $\ell = 1, 2$, be two labels associated with paths ending at the same node in $\mathcal{V}_t^R$. Label $\boldsymbol{\lambda_1}$ is said to dominate label $\boldsymbol{\lambda_2}$ if*

$$\lambda_1^{cost} \leq \lambda_2^{cost} \tag{23}$$
$$\lambda_1^{stp} = \lambda_2^{stp} \tag{24}$$
$$\lambda_1^{time} \leq \lambda_2^{time} \tag{25}$$
$$\lambda_1^{load} \leq \lambda_2^{load} \tag{26}$$
$$\lambda_1^{trip} \leq \lambda_2^{trip} \tag{27}$$
$$\lambda_1^{cust_i} \leq \lambda_2^{cust_i}, \qquad \forall i \in \mathcal{O}, \tag{28}$$

and at least one of the above inequalities is strictly satisfied. If all inequalities are met at equality, then one of the two labels can be declared as dominated by the other but not both.

Dominated labels are discarded. Indeed, conditions (25)–(28) ensure that any feasible extension to the path associated with $\boldsymbol{\lambda_2}$ (the second path) is also feasible for the path represented by $\boldsymbol{\lambda_1}$ (the first path). Furthermore, if (24) is satisfied, then extending the first path with this extension yields a path whose reduced cost is less than or equal to that of the path obtained by extending the second path with the same extension. Note that this property might not be valid if (24) was not imposed because, in this case, the duals used in the resource extension functions (22) depend on the value of λ_k^{stp} and might differ for the same extension.

4.1.2 Shift generation

The reduced cost $\bar{c}_s$ of a shift variable ψ_s, $s \in \mathcal{S}$, is given by

$$\bar{c}_s = c_s + \sum_{p\in\mathcal{P}} \sum_{u\in\mathcal{U}^p} \eta^p f_s^{pu} \pi_{pu}^3 - \pi^6 - \sum_{t\in\mathcal{T}} (b_s^t - e_s^{t-1})\pi_t^8. \tag{29}$$

To generate negative reduced cost variables ψ_s, $s \in \mathcal{S}$, we use a single pricing subproblem that is defined on a network $\mathcal{G}^S = (\mathcal{V}^S, \mathcal{A}^S)$, where $\mathcal{V}^S$ and $\mathcal{A}^S$ denote its node and arc sets, respectively (see Figure 3). The node set $\mathcal{V}^S$ contains: one origin node o representing the start of a shift; one destination node $\overline{o}$ representing the end of a shift; for each product $p \in \mathcal{P}$ and time instant $x \in \mathcal{X}_p$, where $\mathcal{X}_p = \{b^u, e^u \,|\, u \in \mathcal{U}^p\}$, a time node $o_{p,x}^P$ representing time x and product p; and for each time $x \in \{b^u, e^u \,|\, u \in \mathcal{U}\}$, a time node o_x^I representing time x and idleness.

The arc set $\mathcal{A}^S$ contains the following arcs. For each node $o_x^I \in \mathcal{V}^S$ such that x corresponds to the start time of a period, there is an arc (o, o_x^I) to represent the start of a shift. For each node $o_x^I \in \mathcal{V}^S$ such that x corresponds to the end time of a period, there is an arc $(o_x^I, \overline{o})$ to represent the end of a shift. For each node $o_{x_1}^I \in \mathcal{V}^S$ that has an immediate successor node $o_{x_2}^I \in \mathcal{V}^S$ (i.e., x_2 is the end time of the subperiod starting in time x_1), there is an arc $(o_{x_1}^I, o_{x_2}^I)$ representing an idle subperiod starting at time x_1. Similarly, for each product $p \in \mathcal{P}$ and each node $o_{p,x_1}^P \in \mathcal{V}^S$ that has an immediate successor node $o_{p,x_2}^P \in \mathcal{V}^S$, there is an arc $(o_{p,x_1}^P, o_{p,x_2}^P)$ representing the assignment to product p in the subperiod starting at time x_1. For each product $p \in \mathcal{P}$ and each pair of nodes $o_{x_1}^I$ and o_{p,x_2}^P in $\mathcal{V}^S$ such that $x_2 - x_1 = \sigma$ (the duration of a setup), there is an arc $(o_{x_1}^I, o_{p,x_2}^P)$ representing a setup for product p starting at time x_1. Finally, for each product $p \in \mathcal{P}$ and each pair of nodes $o_{p,x}^P$ and o_x^I in $\mathcal{V}^S$, there is an arc $(o_{p,x}^P, o_x^I)$ representing the end of an assignment to product p at time x. To compute the reduced cost and handle the maximum shift length constraint, a number of subperiods y_{jk} and a partial cost c_{jk} are associated with each arc $(j,k) \in A^S$. These

quantities are given in Table 2. In this table, $t^B(x)$ (resp. $t^E(x)$) indicates the period $t \in \mathcal{T}$ such that $b^t = x$ (resp. $e^t = x$), $u^B(x)$ the subperiod $u \in \mathcal{U}$ such that $b^u = x$, and $\hat{\sigma}$ the duration of a setup as a number of subperiods.

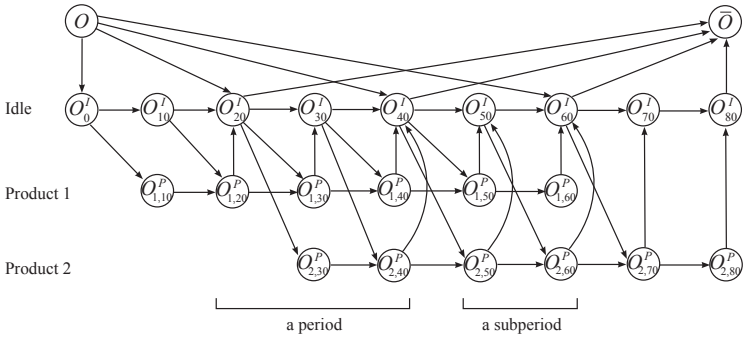

Figure 3: Example of a network $\mathcal{G}^S$ with two products

Table 2: Number of subperiods and partial cost of the arcs in network $\mathcal{G}^S$

arc (j,k)	# subperiods y_{jk}	partial cost c_{jk}
(o, o^I_x)	0	$\gamma^E \underline{l} - \pi^6 - \pi^8_{t^B(x)}$
$(o^I_x, \bar{o})$	0	$\pi^8_{t^E(x)+1}$
$(o^I_{x_1}, o^I_{x_2})$	1	0
$(o^P_{p,x_1}, o^P_{p,x_2})$	1	$\eta^p \pi^3_{p,u^B(x_1)}$
$(o^I_{x_1}, o^P_{p,x_2})$	$\hat{\sigma}$	0
$(o^P_{p,x}, o^I_x)$	0	0

Every feasible shift in $\mathcal{S}$ corresponds to an $o - \bar{o}$ path in $\mathcal{G}^S$ but resource constraints must be imposed to enforce shift feasibility on an arbitrary $o - \bar{o}$ path. Given that network $\mathcal{G}^S$ is acyclic, the pricing subproblem corresponds to a shortest path problem with resource constraints. The following labeling algorithm can be used to solve it.

A partial path from o to a node $j \in \mathcal{V}^S$ is represented by a label $\boldsymbol{\xi} = (\xi^{cost}, \xi^{sp}, \xi^{spl})$, where ξ^{cost} is the path reduced cost, ξ^{sp} is the number of subperiods covered by the path, and ξ^{spl} is the number of subperiods left to reach $\underline{l}$, the minimum number of paid subperiods. Starting from the initial label $\boldsymbol{\xi} = (0, 0, \underline{l})$ associated with node o, the algorithm extends labels forwardly in network $\mathcal{G}^S$. The extension along an arc $(j,k) \in \mathcal{A}^S$ of a label $\boldsymbol{\xi}_j = (\xi^{cost}_j, \xi^{sp}_j, \xi^{spl}_j)$ associated with node j yields a label $\boldsymbol{\xi}_k = (\xi^{cost}_k, \xi^{sp}_k, \xi^{spl}_k)$ whose components are obtained through the following extension functions:

$$\xi^{sp}_k = \xi^{sp}_j + y_{jk} \tag{30}$$

$$\xi^{spl}_k = \max\{\xi^{spl}_j - y_{jk}, 0\} \tag{31}$$

$$\xi^{cost}_k = \xi^{cost}_j + c_{jk} + \gamma^E \delta^S(\xi^{sp}_k, \xi^{sp}_j), \tag{32}$$

where

$$\delta^S(\xi^{sp}_k, \xi^{sp}_j) = \begin{cases} 0 & \text{if } \xi^{sp}_k \leq \underline{l} \\ \xi^{sp}_k - \max\{\xi^{sp}_j, \underline{l}\} & \text{otherwise} \end{cases}$$

indicates the number of subperiods exceeding the minimum number of paid subperiods that must be paid along arc (j,k). Label ξ_k is associated with node k and kept if $\xi^{sp} \leq n^{SP}\, \bar{l}$, where n^{SP} denotes the number of subperiods in a period. Otherwise, it is discarded. The algorithm applies the following dominance rule.

Definition 2 *Let* $\boldsymbol{\xi_\ell} = (\xi_\ell^{cost}, \xi_\ell^{sp}, \xi_\ell^{spl})$, $\ell = 1, 2$, *be two labels associated with paths ending at the same node in* $\mathcal{V}^S$. *Label* $\boldsymbol{\xi_1}$ *is said to dominate label* $\boldsymbol{\xi_2}$ *if*

$$\xi_1^{cost} \leq \xi_2^{cost} \tag{33}$$
$$\xi_1^{sp} \leq \xi_2^{sp} \tag{34}$$
$$\xi_1^{spl} \leq \xi_2^{spl} \tag{35}$$

and at least one of the above inequalities is strictly satisfied. If all inequalities are met at equality, then one of the two labels can be declared as dominated by the other but not both.

4.2 Valid inequalities

To strengthen the linear relaxations encountered throughout the search tree, we try to find valid inequalities violated by the current linear relaxation solution. We consider two families of valid inequalities, namely, subset-row and employee inequalities, which are described below.

The subset-row inequalities, introduced by Jepsen et al. (2008), correspond to Chvátal-Gomory rank-1 inequalities. As in Jepsen et al. (2008), Desaulniers et al. (2008) and several subsequent works, we focus on a particular case of these inequalities that involves only three constraints (2) with multipliers equal to 0.5. The resulting set of inequalities is then given by:

$$\sum_{r \in \mathcal{R}} \left\lfloor \frac{\sum_{i \in \mathcal{O}'} a_{ir}}{2} \right\rfloor \theta_r \leq 1, \qquad \forall \mathcal{O}' \subset \mathcal{O} \quad \text{such that } |\mathcal{O}'| = 3. \tag{36}$$

One such inequality states that, in a feasible solution, there must be at most one route covering at least two of the three customers in $\mathcal{O}'$. Enumeration is used to find violated subset-row inequalities. In a column generation algorithm, these inequalities are added to the master problem but their treatment is not straightforward as the associated dual variables cannot be transferred directly on the arc costs of the route pricing subproblems. An additional component must be added to the labels for each active cut and the dominance must be modified in consequence. The reader is referred to Desaulniers et al. (2011) for complete details.

The employee inequalities impose that a minimum number of production employees be at work during a subset of consecutive periods. They are defined as follows. Let $\mathcal{T}' \subseteq \mathcal{T}$ be a subset of consecutive periods and denote by $t^F(\mathcal{T}')$ and $t^L(\mathcal{T}')$ its first and last periods, respectively. Let $\mathcal{O}' \subseteq \mathcal{O}$ be a subset of orders such that their release times occur during $\mathcal{T}'$. The minimum number of subperiods $\rho(\mathcal{O}')$ required to produce these orders is given by $\rho(\mathcal{O}') = \sum_{p \in \mathcal{P}} \left\lceil \frac{\sum_{i \in \mathcal{O}'} d_i^p}{\eta^p} \right\rceil$. Consequently, to be able to produce all these orders within $\mathcal{T}'$, there must be at least one period in $\mathcal{T}'$ where at least $\kappa(\mathcal{T}', \mathcal{O}') = \left\lceil \frac{\rho(\mathcal{O}')}{n^{SP}\,|\mathcal{T}'|} \right\rceil$ employees are at work. Assuming that the orders in $\mathcal{O}'$ are produced within $\mathcal{T}'$, this constraint can be expressed as:

$$\max_{t \in \mathcal{T}'} w^t \geq \kappa(\mathcal{T}', \mathcal{O}'). \tag{37}$$

This inequality is nonlinear. To linearize it, we propose to weaken the left-hand side by replacing it with $w^{t^F(\mathcal{T}')} + \sum_{s \in \mathcal{S}} \big(\sum_{t \in \mathcal{T}' \setminus \{t^F(\mathcal{T}')\}} b_s^t \big) \psi_s$. This expression counts the number of employees at work in the first period $t^F(\mathcal{T}')$ plus the number of employees that will start a shift within $\mathcal{T}'$. This is a lower bound on $\max_{t \in \mathcal{T}'} w^t$ because some employees may also finish their shifts within $\mathcal{T}'$. Taking into account that some orders in $\mathcal{O}'$ might not have a due time in $\mathcal{T}'$, the employee inequality for $\mathcal{T}'$ and $\mathcal{O}'$ takes the form:

$$w^{t^F(\mathcal{T}')} + \sum_{s \in \mathcal{S}} \Big(\sum_{t \in \mathcal{T}' \setminus \{t^F(\mathcal{T}')\}} b_s^t \Big) \psi_s \geq \kappa(\mathcal{T}', \mathcal{O}') - \sum_{i \in \mathcal{O}'} \kappa_i(\mathcal{T}') \Big(\sum_{\substack{t \in \mathcal{T} \\ t > t^L(\mathcal{T}')}} \sum_{r \in \mathcal{R}} g_{ir}^t \theta_r \Big), \tag{38}$$

where $\kappa_i(\mathcal{T}') = \left\lceil \frac{\rho(\{i\})}{n^{SP}\,|\mathcal{T}'|} \right\rceil$ is an upper bound on the value to be subtracted from $\kappa(\mathcal{T}', \mathcal{O}')$ if order i is removed from $\mathcal{O}'$.

Violated employee inequalities are found by enumeration. For every subset $\mathcal{T}'$ of consecutive periods, we check if the inequality is violated for only one subset of orders $\mathcal{O}'$, namely, that containing the orders with a

release time in $\mathcal{T}'$ and a latest due time in $\mathcal{T}'$ according to the current linear relaxation solution (in this case, $\sum_{i\in\mathcal{O}'} \kappa_i(\mathcal{T}') \sum_{t\in\mathcal{T} \mid t>t^L(\mathcal{T}')} g_{ir}^t\theta_r = 0$ in this solution). The violated employee inequalities found are added to the master problem. Their dual variables can be transferred in the arc costs of the shift network $\mathcal{G}^S$ and be treated like the dual variables of constraints (4) and (14) in the route networks $\mathcal{G}_t^R$, $t \in \mathcal{T}$.

4.3 Branching rules

When branching is required at a node of the search tree, we consider the application of different branching rules. After preliminary tests, we decided to branch on the following entities in this order of priority:

1. Global due times (GDT) as described below;
2. Number of work shifts used;
3. Number of setups for each product;
4. Number of vehicles used in each period;
5. Number of production employees working in each period;
6. Maximum/minimum due time for each order;
7. Arc flow in both network types.

The corresponding branching rules are imposed by adding constraints to the master problem, removing arcs in the subproblem networks, or forbidding the visit to a customer in a route network if it does not satisfy imposed restrictions on its order due time. They can be easily handled by the column generation algorithm. The search tree is explored using a best-first strategy.

As discussed below, evaluating whether a GDT branching rule can be applied at a node of the search tree is time-consuming. In consequence, we do not evaluate it at a node if its evaluation failed at its parent node. Furthermore, we stop evaluating this possibility when reaching a maximum depth in the search tree (15 for our tests).

4.3.1 GDT branching

The production and routing parts of the CPRP are linked together by constraints (4) and (14). Indeed, the delivery trip start times incur production due times for the orders. Earlier due times restrict the production problem and may increase its cost while later due times restrict the routing problem and may also increase its cost. Given that the due times can be identified using binary expressions ($\sum_{r\in\mathcal{R}} g_{ir}^t\theta_r$ for all $t \in \mathcal{T}$ and $i \in \mathcal{O}$), we propose to define a branching rule that separates the solution space using these expressions. Let $\hat{t}_i \in \mathcal{T}_i$ be the period whose start time $b^{\hat{t}_i}$ corresponds to the latest due time used in the current linear relaxation for order $i \in \mathcal{O}$. On one branch, we can impose that, for every order $i \in \mathcal{O}$, its due time be less than or equal to $b^{\hat{t}_i}$ and, on the other branch, we can impose that the due time exceeds $b^{\hat{t}_i}$ for at least one order $i \in \mathcal{O}$. This branching can be expressed as

$$\sum_{i\in\mathcal{O}} \sum_{\substack{t\in\mathcal{T}_i \\ t>\hat{t}_i}} \sum_{r\in\mathcal{R}} g_{ir}^t\theta_r = 0 \qquad \text{and} \qquad \sum_{i\in\mathcal{O}} \sum_{\substack{t\in\mathcal{T}_i \\ t>\hat{t}_i}} \sum_{r\in\mathcal{R}} g_{ir}^t\theta_r \geq 1 \tag{39}$$

for the first and second decisions, respectively. It is valid but not useful in itself because the first decision does not eliminate the current solution. However, given the latest due times $b^{\hat{t}_i}$, $i \in \mathcal{O}$, it might be possible to compute relatively rapidly a lower bound $\underline{z}$ on the production costs with respect to these due times. If $\underline{z}$ is greater than the production costs of the current solution, then the first decision can be enhanced by adding the constraint

$$\sum_{s\in\mathcal{S}} c_s\psi_s \geq \underline{z} \tag{40}$$

to eliminate the current solution. This enhanced branching rule is valid because $\underline{z}$ remains a lower bound on the production costs for due times earlier than $b^{\hat{t}_i}$, $i \in \mathcal{O}$. In consequence, we apply this branching rule only if $\underline{z}$ exceeds the production costs of the current solution.

4.3.2 Lower bound computation

To compute a lower bound $\underline{z}$ on the production costs given the due times $b^{\hat{t}_i}$, $i \in \mathcal{O}$, we propose to solve a MILP that corresponds to a relaxation of the production part of the problem. In this relaxation, we omit the constraint on the maximum shift length and we aggregate the demands of the orders. Given the product release times of each order, we can compute $\overline{M}^{pu}$, the maximum number of units of product $p \in \mathcal{P}$ that can be produced up to subperiod $u \in \mathcal{U}^p$. Similarly, using the due times $b^{\hat{t}_i}$, $i \in \mathcal{O}$, we can also compute $\underline{M}^{pt}$ the minimum number of units of product p that must be produced before the end of period t.

The proposed MILP does not allow the explicit identification of the shifts. It relies on the following five variable types. For each period $t \in \mathcal{T}$, variable w^t indicates the number of employees paid in period t and variable α^t the number of employees starting a shift at the beginning of this period. For each product $p \in \mathcal{P}$ and each subperiod $u \in \mathcal{U}^p$, variable f^{pu} specifies the number of employees assigned to product p in subperiod u while q^{pu} gives the number of units of product p produced in this subperiod. Finally, for each product $p \in \mathcal{P}$ and subperiod $u \in \mathcal{U}$, variable β^{pu} indicates the number of employees starting a setup for product p in subperiod u.

Using this notation, the MILP is:

$$\underline{z} = \min \quad \sum_{t\in\mathcal{T}} \gamma^E n^{SP} w^t + \sum_{p\in\mathcal{P}} \sum_{u\in\mathcal{U}} \gamma^S \beta^{pu} \tag{41}$$

$$\text{s.t.} \quad \sum_{\substack{u\in\mathcal{U}^p \\ e^u \le e^t}} q^{pu} \ge \underline{M}^{pt}, \qquad \forall p \in \mathcal{P}, t \in \mathcal{T} \tag{42}$$

$$\sum_{\substack{u'\in\mathcal{U}^p \\ u' \le u}} q^{pu'} \le \overline{M}^{pu}, \qquad \forall p \in \mathcal{P}, u \in \mathcal{U}^p \tag{43}$$

$$q^{pu} \le \eta^p f^{pu}, \qquad \forall p \in \mathcal{P}, u \in \mathcal{U}^p \tag{44}$$

$$f^{pu} \le \beta^{p,u-\sigma} + f^{p,u-1}, \qquad \forall p \in \mathcal{P}, u \in \mathcal{U}^p \tag{45}$$

$$w^{t(u)} \ge \sum_{p\in\mathcal{P}} \Big(f^{pu} + \sum_{u'\in\mathcal{U}\cap[u-\sigma+1,u]} \beta^{pu'} \Big), \qquad \forall u \in \mathcal{U} \tag{46}$$

$$\alpha^t \ge w^t - w^{t-1}, \qquad \forall t \in \mathcal{T} \tag{47}$$

$$w^t \ge \sum_{j=1}^{\min\{t, \underline{l}/n^{SP}\}} \alpha^{t-j}, \qquad \forall t \in \mathcal{T} \tag{48}$$

$$\sum_{t\in\mathcal{T}} \alpha^t \le n^E \tag{49}$$

$$q^{pu} \ge 0, \qquad \forall p \in \mathcal{P}, u \in \mathcal{U}^p \tag{50}$$

$$f^{pu} \ge 0, \quad \text{integer}, \qquad \forall p \in \mathcal{P}, u \in \mathcal{U}^p \tag{51}$$

$$\beta^{pu} \ge 0, \quad \text{integer}, \qquad \forall p \in \mathcal{P}, u \in \mathcal{U} \tag{52}$$

$$\alpha^t \ge 0, \quad \text{integer}, \qquad \forall t \in \mathcal{T} \tag{53}$$

$$0 \le w^t \le n^W, \quad \text{integer}, \qquad \forall t \in \mathcal{T}. \tag{54}$$

Objective function (41) minimizes the sum of the employee salaries and the setup costs. Constraints (42)–(43) enforce the release and due times for each product. Constraints (44) limit the quantity of each product that can be produced in a subperiod according to the number of employees assigned to this product in this subperiod. Constraints (45) forbid the assignment of an employee to a product in a subperiod except if the production continues from the preceding subperiod or if a setup for this product was just completed. Constraints (46) and (47) allow to compute the number of employees working in each period and the number of employees starting a shift, respectively. Constraints (48) ensure that the employee starting a new shift will be paid for at least $\underline{l}$ periods. Finally, (50)–(54) restrict the feasible domains of the variables. At a given node of the search tree, this model is augmented by the applicable branching decisions.

To compute lower bound $\underline{z}$, we solve MILP (41)–(54) using a commercial MILP solver. However, to limit the total computational time, we impose a limit on the number of nodes that can be explored in the search tree. In any case, the best lower bound obtained at the end of the solution process is assigned to $\underline{z}$.

4.3.3 Lifting

When the GDT branching can be applied for a set of due times $b^{\hat{t}_i}$, $i \in \mathcal{O}$, and a lower bound $\underline{z}$ on the production costs, it might be profitable to enlarge the solution space that will be subject to this lower bound. Indeed, delaying the due time $b^{\hat{t}_i}$ of an order $i \in \mathcal{O}$ might not change the lower bound $\underline{z}$ but it might avoid the exploration of several branch-and-bound nodes in the search tree. To find delayed due times yielding the same lower bound, we propose an iterative greedy procedure based on a modified version of the above MILP (see Algorithm 1). Let N_i, $i \in \mathcal{O}$, be the maximum number of periods by which the due time of order i can be delayed (with respect to $b^{\hat{t}_i}$). The procedure starts at iteration $k = 1$ with the subset of orders $\mathcal{O}^k$ whose due times can be delayed. Then it enters a repeat loop which begins by solving the following MILP that we denote PL^k. For each order $i \in \mathcal{O}^k$, we define N_i binary variables ζ_i^j, $j = 1, \ldots, N_i$. Variable ζ_i^j takes value 1 if the due time of order i is delayed by j periods and 0 otherwise. For $k = 1$, we set $\underline{M}^{ptk} = \underline{M}^{pt}$ for all $p \in \mathcal{P}$ and $t \in \mathcal{T}$. Finally, let ϵ be a sufficiently small positive scalar.

With this notation, MILP PL^k writes as:

$$(PL^k) \qquad \min \quad \sum_{i \in \mathcal{O}^k} \sum_{j=1}^{N_i} j \zeta_i^j \tag{55}$$

$$\text{s.t.} \quad \sum_{t \in \mathcal{T}} \gamma^E n^{SP} w^t + \sum_{p \in \mathcal{P}} \sum_{u \in \mathcal{U}} \gamma^S \beta^{pu} \leq \underline{z} - \epsilon, \tag{56}$$

$$\sum_{\substack{u \in \mathcal{U}^p \\ e^u \leq e^t}} q^{pu} \geq \underline{M}^{ptk} - \sum_{\substack{i \in \mathcal{O}^k \\ \hat{t}_i = t}} d_i^p \Big(\sum_{j=1}^{N_i} \zeta_i^j \Big) + \sum_{\substack{i \in \mathcal{O}^k \\ \hat{t}_i < t \leq \hat{t}_i + N_i}} d_i^p \zeta_i^{t - \hat{t}_i}, \qquad \forall p \in \mathcal{P}, t \in \mathcal{T} \tag{57}$$

$$\sum_{j=1}^{N_i} \zeta_i^j \leq 1, \qquad \forall i \in \mathcal{O}^k \tag{58}$$

$$\zeta_i^j \in \{0, 1\}, \qquad \forall i \in \mathcal{O}^k, j \in \{1, 2, \ldots, N_i\} \tag{59}$$

$$(43) - (54).$$

Objective function (55) aims at minimizing the total number of periods of delay of the orders to obtain production costs that are less than $\underline{z}$ (imposed by constraint (56)). Constraints (57) impose production due times adjusted according to the selected delays. Constraints (58) ensure that at most one delay is chosen for each order and (59) express the binary requirements on the ζ_i^j variables. Finally, constraints (43)–(54) must still be considered.

If PL^k is feasible, then the computed optimal solution indicates for which orders the due times must be delayed and by how many periods to obtain production costs that are less than $\underline{z}$. Given that the number of delayed periods is minimized in PL^k, removing one of these delayed periods yields due times that incur production costs of at least $\underline{z}$. Hence, in Steps 8 and 9, all proposed delays are applied except for one period for one arbitrarily selected order i^* (chosen as the first in $\mathcal{O}^*$ for our tests). Then, the minimum quantities to produce of each product in each period according to the due times are updated in Step 10 before starting a new iteration. The repeat loop stops in two cases: either $\mathcal{O}^*$ becomes empty or PL^k is infeasible in which case the due times of all orders in $\mathcal{O}^k$ can be delayed to their maximum as it is not possible to find a feasible solution yielding a cost less than $\underline{z}$.

Note that this procedure is valid only if PL^k is solved exactly at each iteration. Nevertheless, to avoid long computational times, we limit the number of nodes to explore in the search tree when solving PL^k. When this limit is reached, the procedure is exited immediately without delaying the due times of any additional orders.

Algorithm 1 Lifting procedure for the GDT branching

Set $k := 1$, $STOP := false$, $\mathcal{O}^k = \{i \in \mathcal{O} \mid N_i \geq 1\}$
repeat
  Solve MILP PL^k
  if PL^k is feasible **then**
    $\mathcal{O}^* := \{i \in \mathcal{O}^k \mid \sum_{j=1}^{N_i} \zeta_i^j = 1\}$
    $\forall i \in \mathcal{O}^*$, set $j^*(i) := j$ where $j \in \{1, \ldots, N_i\}$ is such that $\zeta_i^j = 1$
    Choose $i^* \in \mathcal{O}^*$
    Set $\hat{t}_{i^*} := \hat{t}_{i^*} + j^*(i^*) - 1$
    $\forall i \in \mathcal{O}^* \setminus \{i^*\}$, set $\hat{t}_i := \hat{t}_i + j^*(i)$
    $\forall p \in \mathcal{P}, t \in \mathcal{T}$, set $\underline{M}^{pt,k+1} := \underline{M}^{ptk} - \sum_{\substack{i \in \mathcal{O}^k \\ \hat{t}_i = t}} d_i^p \big(\sum_{j=1}^{N_i} \zeta_i^j \big) + \sum_{\substack{i \in \mathcal{O}^k \\ \hat{t}_i < t \leq \hat{t}_i + N_i}} d_i^p \zeta_i^{t - \hat{t}_i}$
    Set $\mathcal{O}^{k+1} := \mathcal{O}^k \setminus \mathcal{O}^*$
    if $\mathcal{O}^* = \emptyset$ **then**
      Set $STOP := true$
  else
    $\forall i \in \mathcal{O}^k$, set $\hat{t}_i := \hat{t}_i + N_i$
    Set $STOP := true$
  $k := k + 1$
until $STOP = true$

5 Computational experiments

To test the performance of the proposed branch-price-and-cut algorithm and its components, we performed a series of computational tests. All these experiments were conducted on a Linux computer equipped with an Intel Xeon processor E3-1226 v3 clocked at 3.3GHz (a single core was used). Moreover, CPLEX 12.4.0.0 was used to solve all restricted master problems and MILPs encountered in the column generation algorithm and the GDT branching technique. When solving MILP (41)–(54), we allow a maximum of 40,000 nodes in the search tree. In the lifting strategy, this limit is reduced to 32,000, 24,000 and 16,000 when solving MILP PL^k (55)–(59), (43)–(54) for the instances with 2, 3 and 4 products, respectively. For all tests, we impose a limit of 7200 seconds (2 hours) on the total computational time.

In this section, we start by describing the set of instances used for our tests. Next, we assess the impact of including the GDT branching with and without the lifting strategy on the total computational time. We also report sensitivity analysis results for other main components of the algorithm. Finally, we study the value of integration, that is, we evaluate what is the benefit of solving the integrated problem rather than solving the production and routing problems separately, in a sequential manner.

5.1 Instances

The instances used for our computational tests are derived from two real-life instances that include around 100 orders and 10 products each and that cannot be handled by the proposed algorithm due to their sizes. In total, we created 108 instances involving 15, 20 or 25 customers. They were obtained as follows. First, we extracted randomly from the available instances 3 subsets of 15 customers, 3 subsets of 20 customers, and 3 subsets of 25 customers. Each subset is used to create 12 different instances: 4 with 2 products, 4 with 3 products, and 4 with 4 products. The desired number of products is obtained from the whole family of products by regrouping the products with the most similar characteristics (product size, production rate, and lifespan). The 4 instances associated with a number of products are obtained by considering 2 demand scenarios and 2 lifespan scenarios. The first demand scenario, denoted o, corresponds to the original demands. In the second demand scenario, denoted d, the original demands are all doubled. As for the lifespan scenarios,

the lifespans range between 150 to 200 minutes for the first one and between 250 and 300 minutes for the second one. These scenarios are simply numbered 1 and 2. Note that the numbers of available workstations, production employees, vehicles and drivers are adjusted for each individual instance. In general, there are 2 workstations and 3 production employees available (resp. 3 and 4) for the instances with demand scenario o (resp. d). The number of vehicles (resp. drivers) available varies between 4 and 6 (resp. 5 and 8).

The parameters of the problem are fixed as reported in Table 3.

Table 3: Parameter values

Parameter	Notation	Value
Period length	$e^t - b^t$, $t \in \mathcal{T}$	30 minutes
Subperiod length	$e^u - b^u$, $u \in \mathcal{U}$	10 minutes
Setup length	σ	10 minutes
Maximum shift and route length	$\bar{l}$	16 periods (8 hours)
Minimum paid time	$\underline{l}$	18 subperiods (3 hours)
Driver salary	γ^D	\$0.25 per minute (\$15/hour)
Employee salary	γ^E	\$2.5 per subperiod (\$15/hour)
Cost of a setup	γ^S	\$1
Vehicle operating costs	γ^V	\$0.25 per minute (\$15/hour)
Preparation time	τ^P	30 minutes
Service time	τ_i^S, $i \in \mathcal{O}$	15 minutes
Maximum number of trips per route	m	2

5.2 Computational results

In this section, we first evaluate the effectiveness of embedding the GDT branching into our branch-price-and-cut algorithm. To do so, we solved the 108 instances using three versions of the algorithm: with the GDT branching, with the GDT branching and the lifting strategy described in Algorithm 1, and without the GDT branching. Table 4 summarizes the results of these experiments by instance class. An instance class is identified by a label nO-mP-vD-wL, where $n \in \{15, 20, 25\}$ specifies the number of orders, $m \in \{2, 3, 4\}$ the number of products, $v \in \{o, d\}$ the demand scenario, and $w \in \{1, 2\}$ the lifespan scenario. Recall that there are 3 instances in each class. For each class and each algorithm, this table reports the number of instances solved to optimality within 7200 seconds, the average computational time in seconds (considering 7200 seconds for each unsolved instance), and the average number of nodes explored in the search tree. The results for each individual instance, including various bounds, are reported in the Appendix. Table 4 includes additional rows that aggregate some results. Rows 15O, 20O, and 25O provide aggregated results over all instances with 15, 20, and 25 orders, respectively. Row ALL provides the results over all instances. Finally, row COMMONLY SOLVED indicates the results for the 42 instances that were solved to optimality by all three algorithms.

The main performance indicator is the number of instances solved to optimality within the time limit. The results in Table 4 clearly show that the GDT branching is very useful. Indeed, a total of 58 instances were solved to optimality with this branching compared to only 43 when it is not used (see row ALL). Incorporating the lifting strategy does not yield improved results (57 instances were solved) although the number of nodes explored decreases on average (8576 nodes with lifting compared to 9133 without it). This general trend is also observed when considering only the instances solved by the three algorithms (row COMMONLY SOLVED). Indeed, the average computational times are similar for both algorithms with GDT branching but much better than without the GDT branching.

As expected, the difficulty of solving an instance increases with the number of products considered. Indeed, for the algorithm with the GDT branching, the number of instances solved to optimality is 23 with 2 products, 21 with 3 products, and 14 with 4 products. Our results also show that the instances with high demands (26 solved to optimality) are slightly more difficult to solve than those with the original demands (32 solved). Higher demands require more resources (drivers, vehicles, employees, workstations), increasing the problem combinatorics. Finally, we also observe that the instances are slightly easier to solve when the lifespans are shorter as it reduces the feasible region. Indeed, 31 instances were solved to optimality for the first lifespan scenario while 27 were solved for the second scenario. Note that similar results are obtained for the other algorithms.

Table 4: Comparative results with and without the GDT branching

Instance class	With GDT			With GDT + Lifting			Without GDT		
	# Opt	Time (s)	# Nodes	# Opt	Time (s)	# Nodes	# Opt	Time (s)	# Nodes
15O-2P-oD-1L	3	7	146	3	19	235	3	48	736
15O-3P-oD-1L	2	2483	21,584	2	2496	21,720	2	2551	20,430
15O-4P-oD-1L	2	2843	49,550	2	2821	45,422	2	2846	22,930
15O-2P-dD-1L	2	2410	50,525	2	2421	38,924	2	2421	42,697
15O-3P-dD-1L	3	2255	15,943	3	2156	13,307	2	2662	15,955
15O-4P-dD-1L	2	2669	15,065	2	2745	20,819	2	4592	24,882
15O-2P-oD-2L	3	37	353	3	112	1081	2	2403	11,190
15O-3P-oD-2L	3	361	1406	3	127	174	2	2422	11,285
15O-4P-oD-2L	3	461	1639	3	465	1628	2	2574	4302
15O-2P-dD-2L	2	2462	5117	2	2448	6219	2	2970	6619
15O-3P-dD-2L	2	4431	1858	1	4800	3029	1	4800	3133
15O-4P-dD-2L	1	4811	6874	1	4808	8046	1	4921	12,398
15O	28	2103	14,172	27	2118	13,384	23	2934	14,713
20O-2P-oD-1L	2	2446	5145	2	2446	6148	2	2445	4371
20O-3P-oD-1L	3	1649	2560	3	1560	2426	1	4800	4014
20O-4P-oD-1L	3	1339	830	2	2548	2819	2	4330	3488
20O-2P-dD-1L	2	2480	14,418	2	2504	18,084	2	2649	18,981
20O-3P-dD-1L	2	2408	5635	2	2408	3471	2	2410	3462
20O-4P-dD-1L	1	4913	6822	1	4943	6228	1	5090	7140
20O-2P-oD-2L	2	2519	5027	2	2629	4713	1	4865	8147
20O-3P-oD-2L	3	2577	1373	2	2958	1431	2	2549	1374
20O-4P-oD-2L	1	6445	2255	1	5869	1312	0	7200	2133
20O-2P-dD-2L	2	3469	6640	3	2969	6136	1	7022	11,132
20O-3P-dD-2L	0	7200	2781	0	7200	2308	0	7200	2523
20O-4P-dD-2L	0	7200	1404	0	7200	906	0	7200	3006
20O	21	3720	4574	20	3770	4665	14	4814	5814
25O-2P-oD-1L	1	6854	22,092	1	6677	17,815	0	7200	24,114
25O-3P-oD-1L	0	7200	15,132	0	7200	18,229	1	4865	12,623
25O-4P-oD-1L	0	7200	13,990	0	7200	13,040	0	7200	14,235
25O-2P-dD-1L	2	3418	7713	2	3383	7428	2	3375	7665
25O-3P-dD-1L	1	7078	10,658	1	7061	8610	0	7200	11,529
25O-4P-dD-1L	0	7200	10,679	0	7200	8689	0	7200	9971
25O-2P-oD-2L	0	7200	8042	2	4976	5769	0	7200	7698
25O-3P-oD-2L	1	6963	3858	0	7200	3166	0	7200	3438
25O-4P-oD-2L	0	7200	2968	0	7200	2717	0	7200	4706
25O-2P-dD-2L	2	4088	4232	2	3707	3088	2	3081	3313
25O-3P-dD-2L	1	5353	3127	1	5872	2787	1	4917	2858
25O-4P-dD-2L	1	7143	1352	1	6084	806	0	7200	3215
25O	9	6408	8654	10	6147	7679	6	6153	8780
ALL	58	4077	9133	57	4012	8576	43	4634	9769
COMMONLY SOLVED	42	429	1044	42	484	977	42	768	2236

Given that the solution process stops when reaching the time limit for a large number of instances, we also investigated the quality of the lower bound reached at the end of the solution process, i.e., at the time limit or before. Table 5 reports for each pair of algorithms the number of best end lower bound reached by each algorithm, excluding the equality cases. For example, columns two and three indicate that, for the 36 instances with 15 customers, the algorithm with the GDT branching obtained a strictly better end lower bound than the algorithm without GDT branching for 12 instances while the opposite is true for only 1 instance. In consequence, both algorithms reached the same end bound for the other 23 instances. These results show again the effectiveness of the GDT branching (much more better end bounds are obtained when using GDT branching) and that the lifting strategy does not enhance much the algorithm.

In the rest of this section, we focus on the algorithm with the GDT branching. With this algorithm, we performed a sensitivity analysis on three of its components: the employee cuts (38), the subset-row cuts (36), and the dynamic generation of the due time constraints (4). To assess these algorithmic components, we considered a varied subset of 12 instances that were solved to optimality by the algorithm with the GDT branching but not easily (computational times ranging between 1211 and 6488 seconds). We selected 4

Table 5: Comparison based on the number of best end lower bounds

# of orders	With GDT	vs	Without GDT	With GDT + Lifting	vs	Without GDT	With GDT + Lifting	vs	With GDT
15	12		1	13		1	3		2
20	17		3	20		1	8		6
25	20		7	22		6	15		10
TOTAL	49		11	55		8	26		18

instances with 15 customers, 4 with 20 customers, and 4 with 25 customers. Each instance was solved three additional times using the algorithm with the GDT branching but without generating employee cuts for the first run, without generating subset-row cuts for the second run, or without generating dynamically due time constraints for the third run (in this case, the due time constraints are all included initially). The results of these experiments are reported in Table 6. The first column gives the instance name, where the first part gives the instance class as above and the last digit indicates the instance number in this class. Then, for each algorithm, including the complete algorithm with GDT branching, we report the computational time obtained for each instance and the number of nodes explored in the search tree. These results clearly show that each of these three components contributes to the efficiency of the algorithm. Indeed, at least 5 instances out of the 12 instances cannot be solved within the 7200-second time limit when one of these components is not activated. In this case, the average computational time increases by at least 29% and most probably by much more given that the solution process was stopped after 7200 seconds for many instances. This time increase is due to a larger number of nodes in the search tree for the first two components (employee and subset-row cuts) and to additional time for solving the linear relaxation for the third component.

Table 6: Sensitivity analysis on some algorithmic components

Instance	With GDT		Less emp. cuts		Less subset-row cuts		Less dynamic DT const.	
	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes	Time (s)	# Nodes
15O-4P-oD-1L-3	1211	5253	2156	11,902	2223	11,902	2364	5185
15O-3P-dD-1L-2	6186	43,767	7104	50,660	7200	50,044	7200	31,347
15O-4P-oD-2L-2	1242	4614	866	4790	883	4790	2698	8498
15O-3P-dD-2L-3	6092	1203	7200	1401	7200	1381	7200	1310
20O-3P-oD-1L-1	4675	6510	7200	10,229	7200	10,249	7200	6184
20O-4P-oD-1L-2	3799	2286	7200	6115	7200	6122	7200	3556
20O-4P-oD-2L-2	4934	479	4763	505	4743	505	4189	621
20O-2P-dD-2L-3	2616	4817	7200	15,075	7200	15,028	4573	4414
25O-2P-oD-1L-1	6162	14,267	7200	10,128	7200	10,204	7200	10,636
25O-2P-dD-1L-2	2861	11,040	4209	11,807	4168	11,807	5606	10,488
25O-3P-oD-2L-1	6488	2995	6726	3225	6555	3225	4466	1723
25O-2P-dD-2L-3	3677	3434	2947	3350	2892	3350	4733	3264
AVERAGE	4161	8389	5398	10,766	5389	10,717	5386	7269

Table 7 reports some statistics on the overall solution process of the algorithm with the GDT branching. Each of the first three rows provides statistics for the instances with the same number of orders that were solved to optimality. The last row gives the same information for all these instances. The given statistics are as follows: the number of instances solved to optimality; the average total computational time in seconds; the average gap in percentage computed using the lower bound obtained before adding cuts and after adding cuts; the average number of cuts generated throughout the search tree for each type of constraints/cuts; the average number of branch-and-bound nodes explored; the average number of times that the GDT branching was applied; and the average percentage of the total time dedicated to solving the MILPs for the GDT branching strategy and to solving the pricing subproblems. Note that the same cut can be generated several times in the search tree, i.e., at different nodes. From these results, we make the following observations. As expected, the average total computational time increases with the number of orders. The gaps are relatively high but the cuts help closing more than 25% of these gaps on average. The average number of relaxed due time constraints (4) that need to be generated is relatively small: 50.1 on average over a total of around 500 such constraints. The average number of times that a GDT branching rule is applied is also relatively small: on average 36.5 for a total of 2398.8 nodes. Given the improved performance that this branching strategy

provides, it shows that every such decision is efficient. Furthermore, the next-to-last column indicates that the proportion of time dedicated to solving MILPs for the GDT branching is controlled (around 17% of the total time). The most time-consuming part of the algorithm is the pricing step that consumes 48.2% of the total time on average. The rest of the computational time is devoted to solving the RMPs.

Table 7: Statistics on the solution process for the instances solved to optimality

# Orders	# Opt.	Time (s)	Gap (%)		Number of cuts			# Nodes	# GDT calls	% of Total time	
			Before cuts	After cuts	Due time	Subset row	Employee			GDT MILP	Pricing
15	28	646.2	5.7	4.0	35.1	755.5	23.2	2628.9	24.5	15.1	46.6
20	21	1235.0	3.7	2.9	53.9	770.4	28.7	1155.8	48.0	21.7	39.2
25	9	4031.1	4.6	3.4	88.0	2158.8	128.4	4583.4	47.1	15.2	55.5
ALL	58	1384.6	4.8	3.5	50.1	978.6	41.5	2398.8	36.5	17.3	48.2

We conducted another series of experiments to evaluate the impact of the length of the periods and the subperiods on the solution process. Instead of considering 30-minute periods and 10-minute subperiods, we considered periods of 15 minutes and subperiods of 5 minutes, increasing the size of model (1)–(15), of the network $\mathcal{G}^S$ for shift generation, and of model (41)–(54) used for the GDT branching strategy. Our tests focused only on the instances with 15 orders. Within the 7200-second time limit, the branch-price-and-cut algorithm with the GDT branching succeeded to solve to optimality only 15 of the 36 instances. This is much less than the 28 solved with larger periods and subperiods.

5.3 Value of integration

In this section, we study the cost reduction that can be achieved by solving the CPRP in an integrated fashion compared to a sequential solution approach currently used by our industrial partner. The sequential approach solves first the routing part of the problem without taking into account the linking constraints with the production. Once the trip departure times are fixed, the production problem is then solved considering the incurred production due times for the orders. For our tests, we used an adapted version of the proposed branch-price-and-cut algorithm with GDT branching for solving each step of the sequential approach.

To make this study, we focus on the 62 instances for which a proven optimal solution was obtained by one of the 3 tested algorithms (with GDT branching, with GDT branching and lifting, without GDT branching). Each of these instances was solved using the sequential approach (except for a few exceptions, they were all solved within one minute of computational time). For 29 of these 62 instances, the sequential approach finds the optimal solution obtained by the integrated approach. More precisely, this occurs for 9 of the 32 instances with the first lifespan scenario and for 20 of the 30 instances with the second lifespan scenarios. This result is not surprising given that lifespans are longer in the second scenario. A cost variation is thus observed for a total of 33 instances (23 with the first lifespan scenario and 10 with the second scenario). In fact, for 6 of these 33 instances, the routing solution obtained in the first step of the sequential approach yielded an infeasible production problem in the second step. To obtain a complete solution for these instances, we incremented iteratively the numbers of workstations and production employees available by one unit each until reaching a feasible production solution.

Table 8 compares the costs of the integrated and the sequential approach solutions for these 33 instances. In this table, we report for each instance the total cost, the production cost and the routing cost for the solution computed by each solution approach. Column xWsEmp indicates the number of extra workstations and production employees required to obtain a solution for the sequential approach if necessary. Finally, the last column provides the cost increase between the two computed solutions. The horizontal lines separate the instances according to the number of orders and to the lifespan scenario. Additional rows 15O, 20O and 25O give average results for the instances with 15, 20 and 25 orders, respectively. Row ALL gives average results over all 33 instances.

From these results, we make the following observations. The 6 instances requiring extra workstations and employees involve the first (tightest) lifespan scenario. For 2 of these instances, more than one extra workstation and employee were needed. For 7 instances, the routing costs are equal in the integrated and the sequential approach solutions, while the sequential approach yields a higher total cost. Given the guaranteed

Table 8: Solution cost comparison between integrated and sequential approaches

Instance	Integrated			Sequential				
	TotalCost	ProdCost	RoutCost	TotalCost	ProdCost	RoutCost	xWsEmp	CostInc
15O-2P-oD-1L-3	350.00	79.00	271.00	365.00	94.00	271.00	0	15.00
15O-3P-oD-1L-1	469.00	159.00	310.00	470.50	174.00	296.50	0	1.50
15O-3P-oD-1L-3	363.00	87.50	275.50	443.00	172.00	271.00	0	80.00
15O-4P-oD-1L-1	469.00	159.00	310.00	469.50	173.00	296.50	0	0.50
15O-4P-oD-1L-3	364.00	88.50	275.50	444.00	173.00	271.00	0	80.00
15O-3P-dD-1L-1	583.75	227.50	356.25	591.75	249.00	342.75	0	8.00
15O-3P-dD-1L-2	597.50	234.00	363.50	659.75	393.50	266.25	4	62.25
15O-3P-dD-1L-3	409.00	133.50	275.50	499.50	226.50	273.00	1	90.50
15O-4P-dD-1L-3	409.00	133.50	275.50	500.75	227.50	273.25	1	91.75
15O-4P-oD-2L-1	401.25	127.00	274.25	403.25	129.00	274.25	0	2.00
15O	441.55	142.85	298.70	484.70	201.15	283.55	0.6	43.15
20O-2P-oD-1L-1	417.50	102.50	315.00	454.00	140.00	314.00	0	36.50
20O-2P-oD-1L-3	550.75	127.00	423.75	588.25	164.50	423.75	0	37.50
20O-3P-oD-1L-1	456.50	142.00	314.50	459.25	145.00	314.25	0	2.75
20O-3P-oD-1L-2	347.75	113.00	234.75	367.75	135.50	232.25	0	20.00
20O-3P-oD-1L-3	578.75	135.50	443.25	616.25	173.00	443.25	0	37.50
20O-4P-oD-1L-1	457.00	143.00	314.00	495.50	181.50	314.00	0	38.50
20O-4P-oD-1L-3	559.25	135.50	423.75	562.25	138.50	423.75	0	3.00
20O-2P-dD-1L-1	463.50	148.50	315.00	471.00	156.00	315.00	0	7.50
20O-2P-dD-1L-2	436.00	148.50	287.50	471.25	186.00	285.25	0	35.25
20O-4P-dD-1L-1	503.75	189.00	314.75	549.50	235.00	314.50	1	45.75
20O-2P-oD-2L-1	404.75	102.50	302.25	412.50	111.00	301.50	0	7.75
20O-3P-oD-2L-1	411.75	104.50	307.25	437.25	135.50	301.75	0	25.50
20O-3P-oD-2L-2	328.00	97.00	231.00	333.75	103.50	230.25	0	5.75
20O-3P-oD-2L-3	515.75	104.50	411.25	526.75	120.50	406.25	0	11.00
20O-4P-oD-2L-2	336.50	112.00	224.50	344.75	121.50	223.25	0	8.25
20O-2P-dD-2L-3	535.75	117.50	418.25	546.25	156.00	390.25	0	10.50
20O	456.45	126.40	330.05	477.27	150.19	327.08	0.06	20.81
25O-2P-oD-1L-1	555.75	165.50	390.25	594.00	204.00	390.00	0	38.25
25O-3P-oD-1L-1	557.75	167.50	390.25	596.00	206.00	390.00	0	38.25
25O-2P-dD-1L-1	610.25	211.50	398.75	686.00	296.00	390.00	1	75.75
25O-3P-dD-1L-1	622.00	221.00	401.00	695.50	305.50	390.00	2	73.50
25O-2P-oD-2L-3	549.75	103.50	446.25	558.25	116.50	441.75	0	8.50
25O-3P-oD-2L-1	514.25	127.00	387.25	525.50	143.00	382.50	0	11.25
25O-2P-dD-2L-3	595.00	148.50	446.50	606.00	163.50	442.50	0	11.00
25O	572.11	163.50	408.61	608.75	204.93	403.82	0.43	36.64
ALL	476.47	139.26	337.21	507.41	177.24	330.17	0.30	30.94

minimum paid time to the drivers, there may exist several routing plans with very similar costs. However, these routes may result in different production schedules. When different routing plans exist with the same cost, the integrated approach is capable of choosing the one incurring the least production cost. For the other instances, the routing costs are always less for the solution obtained by the sequential approach as this solution is computed without regards to the production part of the problem. On the other hand, the production costs are always higher in these solutions. Over these 33 instances (row ALL), when comparing the solutions of the sequential approach to those of the integrated approach, the routing costs decrease by 2.1% on average while the production costs increase by 27.3% on average, resulting in an average total cost increase of 6.5%. We can easily observe that the average total cost increase is much larger for the instances with the first lifespan scenario that makes the linking constraints between routing and production more binding than the second scenario. Recall that this average increase is computed only over the 33 instances where a cost increase was observed. Including the other 29 instances, the average total cost increase would be 3.6%. Nevertheless, this increase is not fully representative given that, for 6 instances, no solution could be found with the sequential approach without augmenting the number of available workstations and production employees. All these results show that it is beneficial to use an integrated approach for solving the CPRP.

6 Conclusions

This paper introduced a new variant of the production-routing problem, called the CPRP, that involves products with short lifespans. The CPRP is highly complex as it includes multiple products, delivery time windows, routes with multiple trips, production employee work shifts to build, and guaranteed minimum paid time for the drivers and the production employees. For this complex problem, we devised an exact branch-price-and-cut algorithm that includes an innovative branching rule. This branching rule imposes a lower bound on the production costs on one branch and forces to delay the due time of at least one order in the other branch. The lower bound is computed by solving a MILP. Computational results on a set of 108 instances derived from two real-life instances and involving between 15 and 25 customers, and 2 to 4 products show that this new branching rule is very efficient. These results also allow to evaluate the value of using an integrated approach for solving the CPRP. Indeed, an average cost increase of 3.6% was observed when using a sequential approach (routing first, production second) to solve the CPRP instead of an integrated approach.

Several future research directions can be proposed, including the following three. First, the proposed branch-price-and-cut algorithm can certainly be improved by developing additional valid inequalities and by finding a way to deal with the very large number of production schedules that yield the same production costs. Also, even if the GDT branching rule revealed to be efficient, using a less time-consuming relaxation for evaluating the lower bound or evaluating with more parsimony the possibility of applying this rule might help reducing the total computational time. Second, the idea behind the GDT branching rule might be applicable to other problems that contain two parts linked only by a small set of constraints. For such problems, it would be interesting to assess if a similar branching rule can be as effective as the GDT branching rule is for the CPRP. Finally, developing an efficient heuristic for solving real-world CPRP instances that can involve up to 150 orders per day is certainly of interest.

Appendix

Detailed comparative results with or without GDT branching

In this appendix, we present the results obtained by the algorithms with the GDT branching, with the GDT branching and the lifting strategy, and without the GDT branching for all 108 tested instances. The name of an instance takes the form nO-mP-vD-wL-x where nO-mP-vD-wL indicates the instance class as described in Section 5.2 and $x \in \{1, 2, 3\}$ gives the instance number in this class. The results are reported in Tables 9–11. In these tables, beside specifying the total computational time and number of branch-and-bound nodes explored for each instance and algorithm, we provide for each instance the lower bound at the root node before adding any cuts (LP bound) and the best upper bound computed in all our tests (Best UB). Furthermore, for each instance and each algorithm, we give the lower bound reached at the end of the solution process (End bound). On the last line of each table, we indicate for each algorithm the average computational time, the average number of nodes explored and the number of instances solved to optimality.

Table 9: Results for instances with 15 customers

Instance	LP bound	Best UB	With GDT			With GDT + Lift			Without GDT		
			Time (s)	# Nodes	End bound	Time (s)	# Nodes	End bound	Time (s)	# Nodes	End bound
15O-2P-oD-1L-1	446.00	462.00	4	48	462.00	8	43	462.00	1	25	462.00
15O-2P-oD-1L-2	430.10	475.75	1	16	475.75	4	19	475.75	11	179	475.75
15O-2P-oD-1L-3	326.62	350.00	16	374	350.00	44	642	350.00	133	2005	350.00
15O-3P-oD-1L-1	447.00	469.00	4	33	469.00	20	37	469.00	84	1054	469.00
15O-3P-oD-1L-2	430.38	478.25	7200	63,424	477.82	7200	63,816	477.82	7200	58,306	476.75
15O-3P-oD-1L-3	336.12	363.00	245	1296	363.00	268	1309	363.00	370	1931	363.00
15O-4P-oD-1L-1	448.00	469.00	118	1318	469.00	64	121	469.00	130	883	469.00
15O-4P-oD-1L-2	426.87	-	7200	142,078	477.75	7200	130,891	477.75	7200	62,916	476.75
15O-4P-oD-1L-3	337.12	364.00	1211	5253	364.00	1198	5253	364.50	1208	4993	364.00
15O-2P-dD-1L-1	499.50	522.00	2	26	522.00	6	26	522.00	3	96	522.00
15O-2P-dD-1L-2	506.11		7200	151,066	546.25	7200	116,095	546.25	7200	127,126	545.25
15O-2P-dD-1L-3	365.12	388.50	29	483	388.50	56	651	388.50	59	870	388.50
15O-3P-dD-1L-1	546.78	583.75	233	2046	583.75	231	2046	583.75	207	2046	583.75
15O-3P-dD-1L-2	525.13	597.50	6186	43,767	597.50	5980	36,691	597.50	7200	42,538	596.75
15O-3P-dD-1L-3	381.30	409.00	346	2016	409.00	258	1185	409.00	578	3281	409.00
15O-4P-dD-1L-1	547.75	577.75	30	288	577.75	102	772	577.75	4259	22,354	577.75
15O-4P-dD-1L-2	505.51	-	7200	40,868	548.25	7200	57,844	548.25	7200	40,684	540.73
15O-4P-dD-1L-3	381.30	409.00	778	4038	409.00	933	3840	409.00	2316	11,608	409.00
15O-2P-oD-2L-1	384.25	384.25	1	10	384.25	1	10	384.25	1	10	384.25
15O-2P-oD-2L-2	375.69	399.75	94	806	399.75	324	3052	399.75	7200	33,429	398.85
15O-2P-oD-2L-3	301.52	326.25	16	242	326.25	11	181	326.25	7	131	326.25
15O-3P-oD-2L-1	393.75	393.75	2	11	393.75	2	11	393.75	2	11	393.75
15O-3P-oD-2L-2	385.05	408.25	816	3764	408.25	333	214	408.25	7200	33,537	407.25
15O-3P-oD-2L-3	310.02	334.75	266	444	334.75	46	298	334.75	66	308	334.75
15O-4P-oD-2L-1	394.75	401.25	47	65	401.25	40	32	401.25	429	885	401.25
15O-4P-oD-2L-2	381.96	408.25	1242	4614	408.25	1264	4614	408.25	7200	11,785	400.75
15O-4P-oD-2L-3	311.02	335.75	93	237	335.75	92	237	335.75	92	237	335.75
15O-2P-dD-2L-1	429.25	430.25	5	27	430.25	3	13	430.25	4	31	430.25
15O-2P-dD-2L-2	434.78	492.50	7200	14,184	477.75	7200	17,786	476.69	7200	14,190	476.50
15O-2P-dD-2L-3	332.31	357.50	182	1139	357.50	141	859	357.50	1706	5635	357.50
15O-3P-dD-2L-1	492.50	492.50	1	10	492.50	1	10	492.50	1	10	492.50
15O-3P-dD-2L-2	453.60	508.50	7200	4360	501.00	7200	8290	507.50	7200	7485	497.31
15O-3P-dD-2L-3	351.12	383.00	6092	1203	383.00	7200	787	376.43	7200	1903	378.49
15O-4P-dD-2L-1	486.00	487.00	33	35	487.00	23	50	487.00	362	1161	487.00
15O-4P-dD-2L-2	446.68	-	7200	15,366	485.75	7200	12,921	485.75	7200	22,288	478.83
15O-4P-dD-2L-3	352.13	-	7200	5222	378.77	7200	11,166	380.16	7200	13,746	378.87
AVERAGE			2103	14,172	# Opt.: 28	2118	13,384	# Opt.: 27	2934	14,713	# Opt.: 23

Table 10: Results for instances with 20 customers

Instance	LP bound	Best UB	With GDT			With GDT + Lift			Without GDT		
			Time (s)	# Nodes	End bound	Time (s)	# Nodes	End bound	Time (s)	# Nodes	End bound
20O-2P-oD-1L-1	412.51	417.50	5	25	417.50	6	25	417.50	6	25	417.50
20O-2P-oD-1L-2	335.75	359.00	7200	14,463	351.19	7200	17,473	351.25	7200	12,141	351.25
20O-2P-oD-1L-3	538.83	550.75	134	947	550.75	132	947	550.75	132	947	550.75
20O-3P-oD-1L-1	420.71	456.50	4675	6510	456.50	3910	5103	456.50	7200	7769	455.25
20O-3P-oD-1L-2	337.75	347.75	242	1010	347.75	767	2144	347.75	7200	4249	346.86
20O-3P-oD-1L-3	563.75	578.75	31	159	578.75	4	30	578.75	2	23	578.75
20O-4P-oD-1L-1	436.29	457.00	65	71	457.00	127	79	457.00	2624	3674	457.00
20O-4P-oD-1L-2	334.25	367.50	3799	2286	367.50	7200	8256	367.50	7200	4586	366.50
20O-4P-oD-1L-3	549.00	559.25	153	133	559.25	317	122	559.25	3167	2204	559.25
20O-2P-dD-1L-1	459.34	463.50	187	987	463.50	237	1079	463.50	660	3871	463.50
20O-2P-dD-1L-2	418.35	436.00	53	179	436.00	76	210	436.00	88	351	436.00
20O-2P-dD-1L-3	562.00	583.75	7200	42,088	581.75	7200	52,962	582.00	7200	52,722	581.75
20O-3P-dD-1L-1	486.68	498.25	22	62	498.25	21	53	498.25	28	68	498.25
20O-3P-dD-1L-2	433.64	436.75	2	10	436.75	2	10	436.75	2	10	436.75
20O-3P-dD-1L-3	593.25	622.50	7200	16,833	615.75	7200	10,351	616.75	7200	10,307	615.75
20O-4P-dD-1L-1	482.31	503.75	338	873	503.75	429	585	503.75	871	1817	503.75
20O-4P-dD-1L-2	406.14	-	7200	5401	424.47	7200	4927	424.31	7200	5857	424.25
20O-4P-dD-1L-3	580.00	603.50	7200	14,192	598.94	7200	13,173	600.00	7200	13,745	597.75
20O-2P-oD-2L-1	390.62	404.75	136	249	404.75	347	193	404.75	7200	12,808	403.53
20O-2P-oD-2L-2	312.35	317.75	220	93	317.75	339	105	317.75	196	53	317.75
20O-2P-oD-2L-3	484.00	513.75	7200	14,739	498.75	7200	13,842	498.75	7200	11,580	497.75
20O-3P-oD-2L-1	395.30	411.75	6950	3915	411.75	7200	3802	411.15	7200	3998	410.00
20O-3P-oD-2L-2	315.99	328.00	686	129	328.00	1588	442	328.00	431	89	328.00
20O-3P-oD-2L-3	505.62	515.75	94	75	515.75	87	48	515.75	15	34	515.75
20O-4P-oD-2L-1	399.14	485.50	7200	3493	416.25	7200	2044	416.00	7200	3708	408.29
20O-4P-oD-2L-2	319.66	336.50	4934	479	336.50	3207	595	336.50	7200	598	335.25
20O-4P-oD-2L-3	480.56	575.75	7200	2793	507.75	7200	1297	507.25	7200	2092	506.25
20O-2P-dD-2L-1	442.40	453.25	592	1262	453.25	3174	8926	453.25	7200	12,153	446.82
20O-2P-dD-2L-2	382.52	412.00	7200	13,840	411.93	2387	4916	412.00	7200	13,621	401.28
20O-2P-dD-2L-3	500.50	535.75	2616	4817	535.75	3347	4566	535.75	6667	7621	535.75
20O-3P-dD-2L-1	463.06	504.75	7200	1981	485.36	7200	2572	486.00	7200	2517	482.16
20O-3P-dD-2L-2	406.52	428.00	7200	1644	421.62	7200	1316	421.65	7200	3302	419.46
20O-3P-dD-2L-3	552.62	-	7200	4718	563.04	7200	3036	565.00	7200	1750	564.25
20O-4P-dD-2L-1	449.60	-	7200	1065	467.21	7200	866	467.06	7200	2894	463.50
20O-4P-dD-2L-2	382.55	-	7200	1994	399.00	7200	1209	399.00	7200	3168	398.50
20O-4P-dD-2L-3	531.63	-	7200	1153	558.92	7200	644	556.97	7200	2957	561.00
AVERAGE			3720	4574	# opt.: 21	3770	4665	# opt.: 20	4814	5814	# opt.: 14

Table 11: Results for instances with 25 customers

Instance	LP bound	Best UB	With GDT			With GDT + Lift			Without GDT		
			Time (s)	# Nodes	End bound	Time (s)	# Nodes	End bound	Time (s)	# Nodes	End bound
25O-2P-oD-1L-1	525.12	555.75	6162	14,627	555.75	5632	9378	555.75	7200	22,126	552.50
25O-2P-oD-1L-2	546.69	-	7200	39,182	569.86	7200	32,373	570.74	7200	32,173	568.03
25O-2P-oD-1L-3	567.17	596.50	7200	12,828	594.12	7200	11,694	594.25	7200	19,043	594.05
25O-3P-oD-1L-1	527.17	557.75	7200	12,179	557.75	7200	12,786	557.75	195	308	557.75
25O-3P-oD-1L-2	548.69	-	7200	18,993	575.43	7200	19,169	571.10	7200	21,504	571.05
25O-3P-oD-1L-3	569.17	597.00	7200	14,224	596.92	7200	22,731	596.75	7200	16,058	596.75
25O-4P-oD-1L-1	529.12	-	7200	8136	554.48	7200	8024	556.33	7200	12,358	551.75
25O-4P-oD-1L-2	544.19	-	7200	19,336	571.17	7200	16,655	570.40	7200	19,363	569.00
25O-4P-oD-1L-3	572.17	653.25	7200	14,499	594.42	7200	14,441	599.62	7200	10,983	596.75
25O-2P-dD-1L-1	586.12	610.25	193	268	610.25	188	268	610.25	185	268	610.25
25O-2P-dD-1L-2	600.19	628.50	2861	11,040	628.50	2760	11,040	628.50	2714	11,040	628.50
25O-2P-dD-1L-3	617.17	-	7200	11,831	645.85	7200	10,975	646.03	7200	11,686	643.39
25O-3P-dD-1L-1	590.12	622.00	6834	6053	622.00	6782	6053	622.00	7200	6959	621.75
25O-3P-dD-1L-2	618.43	-	7200	14,535	639.64	7200	8769	639.03	7200	12,644	636.53
25O-3P-dD-1L-3	631.17	-	7200	11,386	656.50	7200	11,008	656.25	7200	14,984	656.50
25O-4P-dD-1L-1	589.41	-	7200	4830	616.13	7200	5132	615.93	7200	5992	615.07
25O-4P-dD-1L-2	609.15	-	7200	16,355	631.50	7200	12574	631.50	7200	15,206	624.40
25O-4P-dD-1L-3	626.67	-	7200	10,851	647.82	7200	8361	648.30	7200	8714	650.25
25O-2P-oD-2L-1	472.88	500.00	7200	5013	497.75	1338	829	500.00	7200	3029	497.25
25O-2P-oD-2L-2	489.46	-	7200	10,171	508.00	7200	9190	507.61	7200	10,027	506.53
25O-2P-oD-2L-3	522.50	549.75	7200	8941	547.17	6390	7288	549.75	7200	10,038	547.26
25O-3P-oD-2L-1	484.47	514.25	6488	2995	514.25	7200	1691	512.25	7200	1034	510.50
25O-3P-oD-2L-2	497.69	-	7200	4346	522.22	7200	4465	522.93	7200	4061	515.99
25O-3P-oD-2L-3	530.95	555.50	7200	4234	549.41	7200	3341	549.50	7200	5220	555.48
25O-4P-oD-2L-1	484.38	607.00	7200	2611	509.25	7200	1113	512.50	7200	2827	508.81
25O-4P-oD-2L-2	496.77	-	7200	2998	522.00	7200	4576	522.25	7200	4844	510.92
25O-4P-oD-2L-3	531.43	-	7200	3294	547.22	7200	2463	548.66	7200	6446	556.73
25O-2P-dD-2L-1	526.38	546.00	1307	566	546.00	635	362	546.00	222	170	546.00
25O-2P-dD-2L-2	541.27	-	7200	8707	559.13	7200	6110	558.75	7200	8025	554.68
25O-2P-dD-2L-3	572.57	595.00	3677	3434	595.00	3287	2793	595.00	1822	1745	595.00
25O-3P-dD-2L-1	542.67	563.00	1658	518	563.00	3217	623	563.00	350	156	563.00
25O-3P-dD-2L-2	567.88	-	7200	3932	580.25	7200	2701	581.22	7200	5540	580.25
25O-3P-dD-2L-3	581.10	-	7200	4931	602.88	7200	5037	602.72	7200	2879	603.25
25O-4P-dD-2L-1	536.88	557.50	7030	2120	557.50	3853	771	557.50	7200	1861	557.50
25O-4P-dD-2L-2	553.88	-	7200	674	567.23	7200	784	567.23	7200	3692	566.25
25O-4P-dD-2L-3	581.10	-	7200	1262	590.36	7200	864	590.36	7200	4091	602.75
AVERAGE			6408	8654	# opt.: 9	6147	7679	# opt.: 10	6153	8780	# opt.: 6

References

Adulyasak, Y., J.-F. Cordeau, R. Jans. 2014a. Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. INFORMS Journal on Computing **26**(1) 103–120.

Adulyasak, Y., J.-F. Cordeau, R. Jans. 2014b. Optimization-based adaptive large neighborhood search for the production routing problem. Transportation Science **48**(1) 20–45.

Adulyasak, Y., J.-F. Cordeau, R. Jans. 2015. The production routing problem: A review of formulations and solution algorithms. Computers & Operations Research **55**(1) 141–152.

Amorim, P., M.A.F. Belo-Filho, F.M.B. Toledo, C. Almeder, B. Almada-Lobo. 2013. Lot sizing versus batching in the production and distribution planning of perishable goods. International Journal of Production Economics **146**(1) 208–218.

Archetti, C., L. Bertazzi, G. Paletta, M. G. Speranza. 2011. Analysis of the maximum level policy in a production-distribution system. Computers & Operations Research **38**(12) 1731–1746.

Armentano, V.A., A.L. Shiguemoto, A. Løkketangen. 2011. Tabu search with path relinking for an integrated production-distribution problem. Computers & Operations Research **38**(8) 1199–1209.

Armstrong, R., S. Gao, L. Lei. 2008. A zero-inventory production and distribution problem with a fixed customer sequence. Annals of Operations Research **159**(1) 395–414.

Bard, J. F., N. Nananukul. 2008. The integrated production–inventory–distribution–routing problem. Journal of Scheduling **12**(3) 257–280.

Bard, J. F., N. Nananukul. 2009. Heuristics for a multiperiod inventory routing problem with production decisions. Computers & Industrial Engineering **57**(3) 713–723.

Bard, J. F., N. Nananukul. 2010. A branch-and-price algorithm for an integrated production and inventory routing problem. Computers & Operations Research **37**(12) 2202–2217.

Barnhart, C., E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, P. H. Vance. 1998. Branch-and-price: Column generation for solving huge integer programs. Operations Research **46**(3) 316–329.

Belo-Filho, M.A.F., P. Amorim, B. Almada-Lobo. 2015. An adaptive large neighbourhood search for the operational integrated production and distribution problem of perishable products. International Journal of Production Research **53**(20) 6040–6058.

Boudia, M., M.A.O. Louly, C. Prins. 2007. A reactive GRASP and path relinking for a combined production-distribution problem. Computers & Operations Research **34**(11) 3402–3419.

Boudia, M., C. Prins. 2009. A memetic algorithm with dynamic population management for an integrated production-distribution problem. European Journal of Operational Research **195**(3) 703–715.

Chandra, P., M.L. Fisher. 1994. Coordination of production and distribution planning. European Journal of Operational Research **72**(3) 503–517.

Chen, H.-K., C.-F. Hsueh, M.-S. Chang. 2009. Production scheduling and vehicle routing with time windows for perishable food products. Computers & Operations Research **36**(7) 2311–2319.

Chen, Z.-L. 2010. Integrated production and outbound distribution scheduling: Review and extensions. Operations Research **58**(1) 130–148.

Desaulniers, G., J. Desrosiers, S. Spoorendonk. 2011. Cutting planes for branch-and-price algorithms. Networks **58**(4) 301–310.

Desaulniers, G., F. Lessard, A. Hadjar. 2008. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. Transportation Science **42**(3) 387–404.

Frazzon, E.M. 2009. Sustainability and Effectiveness in Global Logistic Systems. GITO-Verlag, Berlin.

Gao, S., L. Qi, L. Lei. 2015. Integrated batch production and distribution scheduling with limited vehicle capacity. International Journal of Production Economics **160**(1) 13–25.

Geismar, H.N., G. Laporte, L. Lei, C. Sriskandarajah. 2008. The integrated production and transportation scheduling problem for a product with a short lifespan. INFORMS Journal on Computing **20**(1) 21–33.

Irnich, S., G. Desaulniers. 2005. Shortest path problems with resource constraints. G. Desaulniers, Jacques Desrosiers, M.M. Solomon, eds., Column Generation, chap. 2. Springer, 33–65.

Jepsen, M., B. Petersen, S. Spoorendonk, D. Pisinger. 2008. Subset-row inequalities applied to the vehicle-routing problem with time windows. Operations Research **56**(2) 497–511.

Lübbecke, M. E., J. Desrosiers. 2005. Selected topics in column generation. Operations Research **53**(6) 1007–1023.

Mula, J., D. Peidro, M. Díaz-Madroñero, E. Vicens. 2010. Mathematical programming models for supply chain production and transport planning. European Journal of Operational Research **204**(3) 377–390.

Schmid, V., K.F. Doerner, G. Laporte. 2013. Rich routing problems arising in supply chain management. European Journal of Operational Research **224**(3) 435–448.

Scholz-Reiter, B., T. Makuschewitz, A.G.N. Novaes, E.M. Frazzon, O.F. Lima Jr. 2011. An approach for the sustainable integration of production and transportation scheduling. International Journal of Logistics Systems and Management **10**(2) 158–179.

Viergutz, C., S. Knust. 2014. Integrated production and distribution scheduling with lifespan constraints. Annals of Operations Research **213**(1) 293–318.