*Les Cahiers du GERAD* ISSN: 0711–2440

**Benders Decomposition for Production Routing under Demand Uncertainty**

Y. Adulyasak, J.-F. Cordeau, R. Jans

G–2012–57

October 2012

# Benders Decomposition for Production Routing under Demand Uncertainty

**Yossiri Adulyasak**
**Jean-François Cordeau**
*GERAD & CIRRELT & HEC Montréal*
*3000 chemin de la Côte-Sainte-Catherine,*
*Montréal (Québec) Canada, H3T 2A7*
yossiri.adulyasak@hec.ca
jean-francois.cordeau@hec.ca

**Raf Jans**
*GERAD & HEC Montréal*
*3000 chemin de la Côte-Sainte-Catherine,*
*Montréal (Québec) Canada, H3T 2A7*
raf.jans@hec.ca

October 2012

*Les Cahiers du GERAD*
G–2012–57

**Abstract**

The production routing problem (PRP) concerns the production and distribution of a single product from a production plant to multiple customers using capacitated vehicles in a discrete and finite time horizon. The PRP is a generalization of the inventory routing problem (IRP) obtained by incorporating production lot-sizing decisions. In this study, we consider the stochastic PRP with demand uncertainty in a two-stage decision process. The decisions in the first stage include production setups and customer visit schedules, while the production and delivery quantities are determined in the second stage. The objective is to jointly minimize the expected total production, inventory, distribution and routing costs. We develop exact solution approaches based on Benders decomposition together with several enhancements for this problem. Two different Benders reformulation schemes are proposed. Furthermore, we compare a standard implementation of the Benders algorithm to one that integrates the Benders cuts within a branch-and-cut framework. This latter implementation is called branch-and-Benders-cut (BBC) and uses only one enumeration tree for the Benders master problem. The Benders master is further enhanced with lower bound lifting inequalities, scenario group cuts and Pareto-optimal cuts. The best version of the Benders decomposition algorithm provides superior results to a branch-and-cut algorithm on the standard formulation when solving a large number of scenarios while the performance of the branch-and-cut algorithm heavily relies on the CPLEX cuts. We further exploit the reoptimization capability of the Benders approach in two stochastic settings, namely, a sample average approximation scheme to handle a large number of scenarios, and a rolling horizon framework for a dynamic and stochastic variant of the PRP. The results show that the computing time of the Benders algorithm using reoptimization is reduced by approximately 40-50% and more than 80% compared to the Benders algorithm without reoptimization and the branch-and-cut algorithm, respectively.

**Key Words:** Integrated supply chain planning; production routing; demand uncertainty; two-stage stochastic problem; Benders decomposition.

**Résumé**

Le problème de production et tournées (PPT) concerne la production et la distribution d'un produit unique à partir d'une usine vers plusieurs clients dans un horizon de planification discret et fini. Le transport se fait à l'aide de véhicules ayant une capacité limitée. Le PPT est une généralisation du problème de tournées de véhicules et de gestion des stocks (PTS) obtenue en incorporant les décisions de lancement et de lotissement pour la production. Dans cette étude, nous considérons le PPT stochastique avec demande incertaine dans un processus de décision en deux étapes. Les décisions de la première étape incluent les lancements de production et le choix des jours de livraison chez les clients, tandis que les décisions liées aux quantités produites et livrées sont déterminées dans la deuxième étape. L'objectif est la minimisation conjointe du coût total espéré de production, stockage et transport. Nous développons des méthodes de résolution exactes basées sur la décomposition de Benders de même que plusieurs raffinements pour ce problème. Deux approches de reformulation de Benders sont proposées. De plus, nous comparons une implémentation classique de la décomposition de Benders à une implémentation qui intègre les coupes de Benders dans un algorithme de séparation et coupes. Cette dernière utilise un seul arbre d'énumération pour le problème maître de Benders. Le problème maître est aussi renforcé par l'ajout d'inégalités valides, de coupes basées sur des groupes de scénarios ainsi que de coupes Pareto-optimales. La meilleure variante de l'algorithme de décomposition de Benders fournit de meilleurs résultats qu'un algorithme classique de séparation et coupes lorsque l'on considère un grand nombre de scénarios. Nous exploitons également la capacité de réoptimisation de la décomposition de Benders dans deux contextes stochastiques, soient une méthode d'échantillonnage pour traiter un grand ensemble de scénarios et un horizon roulant pour une variante dynamique et stochastique du problème. Les résultats montrent que les temps de calcul de la décomposition de Benders utilisant la réoptimisation sont réduits d'environ 40-50% et de plus de 80% lorsque comparés à la décomposition de Benders sans réoptimisation et à un algorithme de séparation et coupe, respectivement.

**Acknowledgments:** This work was supported by the Natural Sciences and Engineering Research Council of Canada under grants 227837-09 and 342182-09. This support is gratefully acknowledged. The authors also thank the RQCHP for providing computing facilities for our experiments.

# 1 Introduction

Demand uncertainty is a major issue in supply chain management as some of the information required for decision making is often known only approximately in the form of forecasts. In these circumstances, solving a deterministic model using point estimates can lead to wrong and costly decisions. One should thus explicitly take the uncertainty into account in the decision process. This study introduces a novel approach to solve the production routing problem (PRP), a generalization of the inventory routing problem (IRP), under demand uncertainty. We consider the problem with a single product in a discrete and finite time horizon where the distribution network consists of a production plant and multiple customers (retailers). Each customer must carry sufficient inventory to satisfy its demand in every period and backlogging is not allowed. The inventory at customers can be replenished by the plant. At the beginning of each period, the plant can perform a setup and make the product using a limited production capacity. The plant can then dispatch a limited number of capacitated vehicles to deliver the product to the customers. If some demand is left unmet at the end of a period, a unit penalty cost has to be paid. This penalty cost can be viewed as the cost of purchasing and delivering the product from an outsourced manufacturer or an opportunity cost associated with the unmet demand. The objective of the problem is to minimize the cost of production, which consists of fixed setup and unit costs, the unit inventory holding cost at both the plant and the customers, the unit cost of unmet demand, and the routing costs of the dispatched vehicles. Note that the PRP reduces to the IRP if the production setup and quantity decisions are fixed. Many studies have addressed the deterministic PRP and IRP where the demand is assumed to be known. Due to their complexity, the majority of the research employed heuristics to solve the problems while only a few studies discussed exact solution approaches (Andersson et al. 2010, Adulyasak et al. 2012b). We refer to Adulyasak et al. (2012a) for a recent review of exact algorithms for the deterministic IRP and PRP.

The difficulty of handling demand uncertainty in the PRP and IRP lies in the fact that it makes the problem intractable (Hvattum and Løkketangen 2009, Solyalı et al. 2012). To the best of our knowledge, no studies have specifically addressed the PRP with demand uncertainty. There are, however, a few studies that have introduced heuristics for the stochastic IRP (SIRP). Federgruen and Zipkin (1984) considered the SIRP with random demands but in a single period planning horizon. The uncertainty is incorporated in the non-linear objective function. The problem is decomposed into an inventory allocation (IA) and a number of traveling salesman problems (TSPs), one for each vehicle. The first subproblem is solved by an exact algorithm and the dual solutions are used to evaluate whether the total cost can be reduced by switching two customers between two different routes. The authors also showed that this approach could provide 6-7% savings compared to the solution obtained by solving a deterministic vehicle routing problem (VRP). A different SIRP involving long term planning horizons was addressed by Jaillet et al. (2002). In this study, a repeated distribution pattern is used and the delivery cost for each customer is fixed. If a stockout occurs, an extra (non-scheduled) delivery is required and a fixed penalty cost is applied. The problem is solved in a rolling horizon framework by using approximations of the direct shipment delivery costs. The IRP with a discrete time infinite horizon was addressed in several studies. Kleywegt et al. (2002a) considered the SIRP with direct deliveries and the problem is represented by a Markov decision process. Since the state space is too large to compute, the authors employed approximate dynamic programming techniques to solve the problem. Kleywegt et al. (2004) extended the previous study to a more general case where a vehicle can deliver to multiple customers in the same route and developed approaches to handle the problem when a vehicle can visit up to three customers in a route. Adelman (2004) focused on the same problem and proposed a price-directed approach where the future costs of current actions are approximated using optimal dual prices. This approach can be used to solve the problem with an unbounded number of customers per route. Hvattum et al. (2009) and Hvattum and Løkketangen (2009) used heuristics based on finite scenario trees to solve the same problem. For the IRP with a discrete finite planning horizon, Bertazzi et al. (2011) addressed the stochastic problem with the order-up-to level (OU) policy and a penalty cost is incurred when a stockout occurs. They employed a heuristic rollout algorithm using an approximate cost-to-go. This approximate cost is formulated as a MIP and is solved by a branch-and-cut algorithm. Solyalı et al. (2012) addressed the single vehicle IRP with demand uncertainty in a discrete finite planning horizon, but the distribution of demand is unknown and backlogging is allowed. This problem is called the robust inventory routing problem (RIRP). They presented robust formulations and used a branch-and-cut algorithm similar to that of Archetti

et al. (2007). Finally, Coelho et al. (2012) considered a dynamic and stochastic variant of the IRP in which demands are gradually revealed over time. They proposed a heuristic to solve the problem and evaluated the value of demand forecasts and transshipments between customers.

In this study, we consider the stochastic PRP (SPRP) under demand uncertainty in a two-stage decision process, where the distribution of the demand is assumed to be known. In the first stage, setup and customer visit decisions, as well as the assignment of vehicles to customers in the case of multiple vehicles must be determined. This is in line with real-world practice, where some decisions such as production setups are decided in advance and these plans remain fixed in order to avoid large disruptions (Hopp and Spearman 2000). Planned visits must be communicated to the customers (and sometimes to the drivers) in advance in order to prepare the workforce, equipment and materials. The replenishment schedules are hence also fixed before the delivery is made. The second stage involves production, inventory and delivery quantity decisions made when the demand becomes known. Since the decisions regarding customer visits and customer-vehicle assignments are made in the first stage, the routing decisions consist of constructing a tour for each vehicle to visit the set of assigned customers. This can be determined in the first stage regardless of the demand realization in the second stage. This problem is an important variant and a practical enhancement to the deterministic PRP and IRP and can also be seen as a generalization of these problems when taking into account the uncertainty of demand. To the best of our knowledge, this problem has not been discussed before. Most of the previous studies on the SIRP consider a Markov decision process where all the decisions are taken independently in each discrete time period and the outcomes of the decisions in each stage affect the subsequent stage. These problems consist of finding optimal decisions in each time period, which lead to the minimal expected total cost in the planning horizon. The only research studying a two-stage decision problem is the RIRP addressed by Solyalı et al. (2012). They focused on robust optimization which attempts to find a solution that is immune to any realization of demand. Their approach is appropriate when the distribution of demand is unknown. However, since information such as historical demands is typically available, one can construct demand profiles and use a two-stage stochastic model to solve the problem. Moreover, both the SPRP and SIRP concern operational planning where the decision process is repeated a large number of times over a long horizon. Instead of using robust optimization to find the solution with the minimal worst case cost, one can possibly benefit to a greater degree in the long term by using the two-stage stochastic programming approach to find the solution where the total expected operating cost is minimized.

The main contributions of our study are fourfold. First, we introduce the two-stage stochastic PRP and provide a standard formulation which is an extension of the deterministic PRP formulation presented by Adulyasak et al. (2012a). Second, we propose two Benders reformulations for the SPRP. In the first decomposition, the second-stage quantity decision variables are projected out, while the routing variables are also projected out in the second decomposition. Third, we develop exact algorithms for the Benders reformulations based on a branch-and-cut framework, called branch-and-Benders-cut (BBC), where the Benders cuts are used in conjunction with subtour elimination constraints. This algorithm is then compared to a classical Benders algorithm. Several computational enhancements are proposed: lower bound lifting inequalities, scenario group cuts and Pareto-optimal cuts, and extensive computational results are provided. A comparison with an extension of the successful branch-and-cut (BC) approach for the deterministic problems (Archetti et al. 2007, 2011, Solyalı and Süral 2011, Adulyasak et al. 2012a) indicates the superiority of the best version of the BBC approach using the first decomposition for problems with a large number of scenarios. Fourth, we demonstrate the benefits of the reoptimization capabilities of Benders decomposition for the SPRP in two stochastic environments, i.e., in a sample average approximation method (SAA) and in a rolling horizon (RH) framework. In these two settings, we obtain improvements in CPU time of 50% (for SAA) and 41% (for RH) compared to the Benders approach without reoptimization, and of 83% (for SAA) and 81% (for RH) compared to the branch-and-cut approach.

The rest of the paper is organized as follows. Section 2 presents notation and formulations for the SPRP. Section 3 describes the Benders decomposition approaches to solve the problem. The details of the solution algorithms are provided in Section 4. This is followed by the computational experiments in Sections 5 and 6, and by the conclusion.

# 2 SPRP Formulations

## 2.1 Notation

Let $\Omega$ denote the finite set of demand scenarios (this set can be enumerated explicitly if the number of possible demand realizations is not too large or it can be constructed by a sampling method). Let also $\omega \in \Omega$ be the index of the demand scenarios. The two-stage SPRP can be defined on a complete undirected graph $G = (N, E)$ with the following notations.

Sets:

$T$ set of time periods, indexed by $t \in \{1, \ldots, l\}$;

$N$ set of plant and customers, indexed by $i \in \{0, \ldots, n\}$, where the plant is represented by node 0 and $N_c = N \setminus \{0\}$ is the subset of $n$ customers;

$E$ set of edges, $E = \{(i, j) : i, j \in N, i < j\}$;

$K$ set of identical vehicles, indexed by $k \in \{1, \ldots, m\}$;

$E(S)$ set of edges $(i, j) \in E$ such that $i, j \in S$, where $S \subseteq N$ is a given set of nodes;

$\delta(S)$ set of edges incident to a node set $S$, $\delta(S) = \{(i, j) \in E : i \in S, j \notin S \text{ or } i \notin S, j \in S\}$ (for simplicity, we also write $\delta(i)$ to represent the set $\delta(\{i\})$ of edges incident to node $i$).

Decision variables:

$y_t$ equal to 1 if production takes place in period $t$, 0 otherwise;

$z_{ikt}$ equal to 1 if node $i$ is visited by vehicle $k$ in period $t$, 0 otherwise;

$x_{ijkt}$ if vehicle $k$ travels directly between node $i$ and node $j$ in period $t$, 0 otherwise;

$p_{t\omega}$ production quantity in period $t$ under scenario $\omega$;

$I_{it\omega}$ inventory at node $i$ at the end of period $t$ under scenario $\omega$;

$q_{ikt\omega}$ quantity delivered to customer $i$ with vehicle $k$ in period $t$ under scenario $\omega$;

$e_{it\omega}$ amount of unmet demand at customer $i$ in period $t$ associated with scenario $\omega$.

Parameters:

$u$ unit production cost;

$f$ fixed production setup cost;

$h_i$ unit inventory holding cost at node $i$;

$c_{ij}$ transportation cost between nodes $i$ and $j$;

$\sigma_i$ unit cost of unmet demand of customer $i$;

$C$ production capacity;

$Q$ vehicle capacity;

$L_i$ maximum or target inventory level at node $i$;

$d_{it\omega}$ demand of customer $i$ in period $t$ under scenario $\omega$ (we assume throughout that $d_{it\omega} \leq L_i, \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega$);

$\rho_\omega$ probability of scenario $\omega$;

$I_{i0}$ initial inventory available at node $i$.

Let also $I_{i0\omega} = I_{i0}, \forall \omega \in \Omega$, $M_{t\omega} = \min\left\{C, \sum_{j=t}^{l} \sum_{i \in N_c} d_{ij\omega}\right\}$ and $M'_{it\omega} = \min\left\{L_i, Q, \sum_{j=t}^{l} d_{ij\omega}\right\}$.

## 2.2 Two-Stage SPRP Formulation

We first present a two-stage SPRP formulation which is an extension of the formulation used in the branch-and-cut approaches for the deterministic problems studied by Archetti et al. (2007, 2011), Solyalı and Süral (2011) and Adulyasak et al. (2012a). The SPRP can be formulated as follows:

$$\min \sum_{t \in T} \left( f y_t + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} x_{ijkt} + \sum_{\omega \in \Omega} \rho_\omega \left( u p_{t\omega} + \sum_{i \in N} h_i I_{it\omega} + \sum_{i \in N_c} \sigma_i e_{it\omega} \right) \right) \tag{1}$$

s.t.
$$I_{0,t-1,\omega} + p_{t\omega} = \sum_{i\in N_c}\sum_{k\in K} q_{ikt\omega} + I_{0t\omega} \qquad \forall t \in T, \forall \omega \in \Omega \quad (2)$$
$$I_{i,t-1,\omega} + \sum_{k\in K} q_{ikt\omega} + e_{it\omega} = d_{it\omega} + I_{it\omega} \qquad \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega \quad (3)$$
$$I_{0t\omega} \leq L_0 \qquad \forall t \in T, \forall \omega \in \Omega \quad (4)$$
$$I_{it\omega} + d_{it\omega} \leq L_i \qquad \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega \quad (5)$$
$$p_{t\omega} \leq M_{t\omega} y_t \qquad \forall t \in T, \forall \omega \in \Omega \quad (6)$$
$$\sum_{i\in N_c} q_{ikt\omega} \leq Q z_{0kt} \qquad \forall k \in K, \forall t \in T, \forall \omega \in \Omega \quad (7)$$
$$q_{ikt\omega} \leq M'_{it\omega} z_{ikt} \qquad \forall i \in N_c, \forall k \in K, \forall t \in T, \forall \omega \in \Omega \quad (8)$$
$$\sum_{k\in K} z_{ikt} \leq 1 \qquad \forall i \in N_c, \forall t \in T \quad (9)$$
$$\sum_{(j,j')\in\delta(i)} x_{jj'kt} = 2 z_{ikt} \qquad \forall i \in N, \forall k \in K, \forall t \in T \quad (10)$$
$$\sum_{(i,j)\in E(S)} x_{ijkt} \leq \sum_{i\in S} z_{ikt} - z_{ekt} \qquad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall k \in K, \forall t \in T \quad (11)$$
$$p_{t\omega}, I_{it\omega}, q_{ikt\omega} \geq 0 \qquad \forall i \in N, \forall k \in K, \forall t \in T, \forall \omega \in \Omega \quad (12)$$
$$y_t, z_{ikt} \in \{0,1\} \qquad \forall i \in N, \forall k \in K, \forall t \in T \quad (13)$$
$$x_{ijkt} \in \{0,1\} \qquad \forall (i,j) \in E : i \neq 0, \forall k \in K, \forall t \in T \quad (14)$$
$$x_{0jkt} \in \{0,1,2\} \qquad \forall j \in N_c, \forall k \in K, \forall t \in T. \quad (15)$$

The objective function (1) minimizes the total cost of the first stage decisions and the expected total cost of the second stage decisions. Constraints (2) and (3) control the inventory flow balance for each scenario at the plant and customers, respectively. The maximum inventory level at the plant and customers is imposed by constraints (4) and (5), respectively. Constraints (6) allow a positive production quantity only if a setup is made; this quantity cannot exceed the minimum of the production capacity and the total demand in the remaining periods. The delivery quantity in each vehicle cannot exceed the vehicle capacity imposed by constraints (7) and a positive delivery quantity is allowed only if the customer is visited according to constraints (8). Each customer cannot be visited more than once per period following constraints (9). Constraints (10) require the number of edges incident to be 2 if the node is visited and constraints (11) are subtour elimination constraints (SECs) for each vehicle. We remark that constraints (5) impose the inventory capacity at customers when the delivery is made and prior to demand consumption. These constraints can also be written as $I_{i,t-1,\omega} + \sum_{k\in K} q_{ikt\omega} + e_{it\omega} \leq L_i$. Note that this formulation becomes the vehicle index formulation for the deterministic PRP in Adulyasak et al. (2012a) when the number of scenarios is equal to one and the extra delivery quantity is forced to be zero.

To strengthen the routing part, the following inequalities can be added to the formulation (see Archetti et al. 2007, 2011):
$$z_{ikt} \leq z_{0kt} \qquad \forall i \in N_c, \forall k \in K, \forall t \in T \quad (16)$$
$$x_{ijkt} \leq z_{ikt} \text{ and } x_{ijkt} \leq z_{jkt} \qquad \forall (i,j) \in E(N_c), \forall k \in K, \forall t \in T. \quad (17)$$

Constraints (16) allow a vehicle to visit customers only if it is dispatched from the plant and constraints (17) allow an edge incident to a customer node only if that customer is visited.

When addressing the multi-vehicle aspect, Adulyasak et al. (2012a) showed that the following valid vehicle symmetry breaking constraints can significantly improve the performance of the branch-and-bound process:
$$z_{0kt} \geq z_{0,k+1,t} \qquad 1 \leq k \leq m-1, \forall t \in T \quad (18)$$
$$\sum_{i=1}^{j} 2^{(j-i)} z_{ikt} \geq \sum_{i=1}^{j} 2^{(j-i)} z_{i,k+1,t} \qquad \forall j \in N_c \quad 1 \leq k \leq m-1, \forall t \in T. \quad (19)$$

Model (1)–(19) will be referred to as the basic formulation ($BF$).

A simple modification can be made to formulate the two-stage SIRP where the production quantity in each period is fixed. Denote by $B_t$ the production quantity made available in each period. One can set all the production setup variables $y_t$ to one ($y_t = \bar{y}_t = 1$). Constraints (6) and the parameters $M'_{it\omega}$ are replaced with $p_{t\omega} = B_t\bar{y}_t, \forall t \in T, \forall \omega \in \Omega$ and $\min\{L_i, Q\}$, respectively. The rest of the formulation remains unchanged.

# 3 Benders Decomposition Approaches

We now introduce exact algorithms based on the Benders decomposition scheme (Benders 1962) to solve the two-stage SPRP. In Benders decomposition, the original problem is reformulated into a *master problem*, and a number of *subproblems* which are typically easier to solve than the original problem. By using linear programming duality, all the variables that belong to the subproblems are projected out and the master problem contains the remaining variables and an artificial variable representing a lower bound on the cost of the subproblem. In the original concept of Benders decomposition, the problem is solved by a cutting plane algorithm. At each iteration, the values of the master problem variables are first determined and the subproblems are solved by holding these variables fixed. If the subproblems are feasible and bounded, an *optimality cut* is added to the master problem, otherwise a *feasibility cut* is added. An upper bound can be computed from the subproblems and a lower bound is obtained if the master problem is solved to optimality. The process continues until an optimal solution is found or the optimality gap is smaller than a given threshold value.

In the two-stage SPRP, we observe that the integer variables are the first-stage decisions while the continuous variables belong to the second-stage. If the decisions in the first stage are fixed, the resulting subproblem is a network flow problem which can be decomposed by scenario. This follows the original idea of applying Benders decomposition to stochastic integer programs, also known as the $L$-shaped method (see Van Slyke and Wets 1969, Birge and Louveaux 2011). In this section, we present two different Benders decomposition schemes. The first one projects out the flow variables of the second stage, i.e., production, inventory, delivery and unmet demand quantities, while the second one also projects out the routing variables.

## 3.1 Benders Reformulation 1 (BR1)

The first Benders reformulation is constructed by separating the first-stage and second-stage decisions into a master problem and subproblems, respectively. We let $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$ and $\bar{\mathbf{z}}$ denote the vectors of fixed $x_{ijkt}$, $y_t$ and $z_{ikt}$ variables, respectively. One can observe that the second-stage decisions are independent of $\bar{\mathbf{x}}$. The expected total cost of the second-stage decisions, denoted by $\upsilon(\bar{\mathbf{y}}, \bar{\mathbf{z}})$, can be calculated as $\upsilon(\bar{\mathbf{y}}, \bar{\mathbf{z}}) = \sum_{\omega\in\Omega} \rho_\omega \upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$, where $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$ is the total second-stage cost of scenario $\omega$ which can itself be obtained by solving the following *primal flow subproblem (PFS)*:

$$\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}}) = \min \sum_{t\in T} \left( up_{t\omega} + \sum_{i\in N} h_i I_{it\omega} + \sum_{i\in N_c} \sigma_i e_{it\omega} \right) \tag{20}$$

s.t. (12) and

$$I_{0,t-1,\omega} + p_{t\omega} = \sum_{i\in N_c} \sum_{k\in K} q_{ikt\omega} + I_{0t\omega} \qquad \forall t \in T \tag{21}$$

$$I_{i,t-1,\omega} + \sum_{k\in K} q_{ikt\omega} + e_{it\omega} = I_{it\omega} + d_{it\omega} \qquad \forall i \in N_c, \forall t \in T \tag{22}$$

$$I_{0t\omega} \leq L_0 \qquad \forall t \in T \tag{23}$$

$$I_{it\omega} + d_{it\omega} \leq L_i \qquad \forall i \in N_c, \forall t \in T \tag{24}$$

$$p_{t\omega} \leq M_{t\omega}\bar{y}_t \qquad \forall t \in T \tag{25}$$

$$\sum_{i\in N_c} q_{ikt\omega} \leq Q\bar{z}_{0kt} \qquad \forall k \in K, \forall t \in T \tag{26}$$

$$q_{ikt\omega} \leq M'_{it\omega} \bar{z}_{ikt} \qquad \forall i \in N_c, \forall k \in K, \forall t \in T. \tag{27}$$

Due to the presence of the variables $e_{it\omega}$, the PFS is always feasible because the demand can be left unmet. Furthermore, since the cost parameters $u$, $h_i$ and $\sigma_i$ are finite and due to constraints (21)–(24), any feasible solution of the PFS must be bounded. We let $\boldsymbol{\alpha} = (\alpha_{t\omega} | \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\beta} = (\beta_{it\omega} | \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\gamma} = (\gamma_{t\omega} \geq 0 | \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\theta} = (\theta_{it\omega} \geq 0 | \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\delta} = (\delta_{t\omega} \geq 0 | \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\kappa} = (\kappa_{kt\omega} \geq 0 | \forall k \in K, \forall t \in T, \forall \omega \in \Omega)$ and $\boldsymbol{\zeta} = (\zeta_{ikt\omega} \geq 0 | \forall i \in N_c, \forall k \in K, \forall t \in T, \forall \omega \in \Omega)$ be the vectors of the dual variables associated with constraints (21)–(27), respectively, and let also $\alpha_{l+1,\omega} = 0$ and $\beta_{i,l+1,\omega} = 0$. The dual of the primal subproblem for each scenario $\omega$, called the *dual flow subproblem (DFS)*, can be formulated as follows.

$$\begin{aligned} \upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}}) = \max \quad & -I_{00}\alpha_{1\omega} + \sum_{i \in N_c} (d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i \in N_c} \sum_{t=2}^{l} d_{it\omega}\beta_{it\omega} - \sum_{t \in T} L_0 \gamma_{t\omega} \\ & - \sum_{t \in T} \sum_{i \in N_c} (L_i - d_{it\omega})\theta_{it\omega} - \sum_{t \in T} M_{t\omega} \bar{y}_t \delta_{t\omega} - \sum_{t \in T} \sum_{k \in K} Q \bar{z}_{0kt} \kappa_{kt\omega} - \sum_{t \in T} \sum_{k \in K} \sum_{i \in N_c} M'_{it\omega} \bar{z}_{ikt} \zeta_{ikt\omega} \end{aligned} \tag{28}$$

$$\text{s.t.} \quad \alpha_{t\omega} - \delta_{t\omega} \leq u \qquad \forall t \in T \tag{29}$$

$$-\alpha_{t\omega} + \alpha_{t+1,\omega} - \gamma_{t\omega} \leq h_0 \qquad \forall t \in T \tag{30}$$

$$-\beta_{it\omega} + \beta_{i,t+1,\omega} - \theta_{it\omega} \leq h_i \qquad \forall i \in N_c, \forall t \in T \tag{31}$$

$$-\alpha_{t\omega} + \beta_{it\omega} - \kappa_{kt\omega} - \zeta_{ikt\omega} \leq 0 \qquad \forall i \in N_c, \forall k \in K, \forall t \in T \tag{32}$$

$$\beta_{it\omega} \leq \sigma_i \qquad \forall i \in N_c, \forall t \in T. \tag{33}$$

Let $\Delta_\omega$ denote the polyhedron defined by constraints (29)–(33). Since the PFS is always feasible and bounded, by strong duality, the DFS is feasible and bounded. We further let $\Delta = \bigcup_{\omega \in \Omega} \Delta_\omega$ and $P_\Delta$ be the set of extreme points of $\Delta$.

To formulate the *Benders master problem*, we define $\pi_\omega(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}) = -I_{00}\alpha_{1\omega} + \sum_{i \in N_c}(d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i \in N_c} \sum_{t=2}^{l} d_{it\omega}\beta_{it\omega} - \sum_{t \in T} L_0 \gamma_{t\omega} - \sum_{t \in T} \sum_{i \in N_c} (L_i - d_{it\omega})\theta_{it\omega}$ and we introduce an artificial variable $\eta$ representing the expected total flow cost. The original model (1)–(15) can be reformulated as follows.

$$\min \sum_{t \in T} \left( f y_t + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} x_{ijkt} + \eta \right) \tag{34}$$

s.t. (9)–(11), (13)–(15) and

$$\sum_{\omega \in \Omega} \rho_\omega \left( - \sum_{t \in T} M_{t\omega} \delta_{t\omega} y_t - \sum_{t \in T} \sum_{k \in K} Q \kappa_{kt\omega} z_{0kt} - \sum_{t \in T} \sum_{k \in K} \sum_{i \in N_c} M'_{it\omega} \zeta_{ikt\omega} z_{ikt} + \pi_\omega(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}) \right) \leq \eta$$
$$\forall (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}, \boldsymbol{\delta}, \boldsymbol{\zeta}, \boldsymbol{\kappa}) \in P_\Delta. \tag{35}$$

This formulation, together with the valid inequalities (16)–(19), will be referred to as BMP1.

Observe that BMP1 contains a large number of Benders cuts (35) as well as the SECs (11). Thus, a natural solution approach is to start from a relaxed BMP1 where these constraints are dropped. Next, violated constraints are detected and iteratively added to the problem. The solution approaches to handle this reformulation will be explained in Section 4.

### 3.2 Benders Reformulation 2 (BR2)

Due to the presence of the SECs (11) in BMP1, the Benders master problem cannot be solved by a standard branch-and-bound procedure as is done in a typical Benders decomposition algorithm. One alternative is

to reformulate the problem by keeping only the $y_t$ and $z_{ikt}$ variables in the master problem. The routing variables $x_{ijkt}$ and the associated constraints yield the following subproblem:

$$r(\bar{\mathbf{z}}) = \min \sum_{t\in T} \left( \sum_{(i,j)\in E} \sum_{k\in K} c_{ij} x_{ijkt} \right) \tag{36}$$

s.t. (14)–(15) and

$$\sum_{(j,j')\in\delta(i)} x_{jj'kt} = 2\bar{z}_{ikt} \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{37}$$

$$\sum_{(i,j)\in E(S)} x_{ijkt} \leq \sum_{i\in S} \bar{z}_{ikt} - \bar{z}_{ekt} \qquad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall k \in K, \forall t \in T. \tag{38}$$

This subproblem can be separated by vehicle and time period, and reduces to independent TSPs. The number of routing subproblems that must be solved is then equal to the number of vehicles dispatched during the planning horizon. Denote by $N_c^{kt}(\bar{\mathbf{z}})$ the set of customers visited by vehicle $k$ in period $t$ corresponding to the fixed decision vector $\bar{\mathbf{z}}$, i.e., $N_c^{kt}(\bar{\mathbf{z}}) = \{i|\bar{z}_{ikt} = 1\}$ and $N^{kt}(\bar{\mathbf{z}}) = N_c^{kt}(\bar{\mathbf{z}}) \cup \{0\}$. This subproblem can be rewritten as $r(\bar{\mathbf{z}}) = \sum_{t\in T} \sum_{k\in K} r_{kt}(\bar{\mathbf{z}})$, where $r_{kt}(\bar{\mathbf{z}})$ is the total routing cost of vehicle $k$ in period $t$, which can be determined by solving the following problem:

$$r_{kt}(\bar{\mathbf{z}}) = \min \sum_{(i,j)\in E} c_{ij} x_{ijkt} \tag{39}$$

s.t. (14)–(15) and

$$\sum_{(j,j')\in\delta(i)} x_{jj'kt} = 2\bar{z}_{ikt} \qquad \forall i \in N \tag{40}$$

$$\sum_{(i,j)\in E(S^{kt})} x_{ijkt} \leq |S^{kt}| - 1 \qquad \forall S^{kt} \subseteq N_c^{kt}(\bar{\mathbf{z}}) : |S^{kt}| \geq 2. \tag{41}$$

To dualize this model, we relax the integrality constraints (14) and (15) by replacing them with the following constraints:

$$0 \leq x_{ijkt} \leq 1 \qquad \forall (i,j) \in E : i \neq 0 \tag{42}$$

$$0 \leq x_{0jkt} \leq 2 \qquad \forall j \in N_c. \tag{43}$$

Model (39) and (40)–(43) will be referred to as the *primal routing subproblem (PRS)*.

Denote by $\boldsymbol{\varphi} = (\varphi_{ikt}|\forall i \in N, \forall k \in K, \forall t \in T)$, $\boldsymbol{\vartheta} = (\vartheta_{S^{kt}} \geq 0|\forall k \in K, \forall t \in T, \forall S^{kt} \subseteq N_c^{kt}(\bar{\mathbf{z}}) : |S^{kt}| \geq 2)$, $\boldsymbol{\mu} = (\mu_{ijkt} \geq 0|\forall (i,j) \in E : i \neq 0, \forall k \in K, \forall t \in T)$ and $\boldsymbol{\mu}^0 = (\mu^0_{jkt} \geq 0|\forall j \in N_c, \forall k \in K, \forall t \in T)$ the vectors of the dual variables associated with constraints (40)–(43), respectively. The *dual routing subproblem (DRS)* can be formulated as follows.

$$r_{kt}(\bar{\mathbf{z}}) = \max \sum_{i\in N} 2\bar{z}_{ikt}\varphi_{ikt} - \sum_{S^{kt}\subseteq N_c^{kt}(\bar{\mathbf{z}}):|S^{kt}|\geq 2} (|S^{kt}| - 1)\vartheta_{S^{kt}} - \sum_{(i,j)\in E:i\neq 0} \mu_{ijkt} - 2\sum_{j\in N_c} \mu^0_{jkt} \tag{44}$$

s.t.

$$\varphi_{ikt} + \varphi_{jkt} - \sum_{S^{kt}:i,j\in S^{kt}} \vartheta_{S^{kt}} - \mu_{ijkt} \leq c_{ij} \qquad \forall (i,j) \in E : i \neq 0 \tag{45}$$

$$\varphi_{ikt} + \varphi_{jkt} - \sum_{S^{kt}:i,j\in S^{kt}} \vartheta_{S^{kt}} - \mu^0_{jkt} \leq c_{0j} \qquad \forall (i,j) \in E : i = 0. \tag{46}$$

Denote by $\boldsymbol{\Theta}_{kt}$ the polyhedron defined by constraints (45)–(46). Since the cost $c_{ij}$ is finite, both the PRS and DRS are feasible and bounded. Let also $\boldsymbol{\Theta} = \bigcup_{k\in K, t\in T} \boldsymbol{\Theta}_{kt}$ and $P_{\boldsymbol{\Theta}}$ be the set of extreme points of $\boldsymbol{\Theta}$. To

formulate the Benders master problem, we introduce an artificial variable $\tau$ representing the total routing cost and we define $\psi_{kt}(\boldsymbol{\vartheta}, \boldsymbol{\mu}, \boldsymbol{\mu^0}) = -\sum_{S^{kt} \subseteq N_c^{kt}(\bar{\mathbf{z}}):|S^{kt}|\geq 2}(|S^{kt}|-1)\vartheta_{S^{kt}} - \sum_{(i,j)\in E: i\neq 0} \mu_{ijkt} - 2\sum_{j\in N_c} \mu^0_{jkt}$. The following Benders master problem is obtained:

$$\min \sum_{t\in T}(fy_t + \eta + \tau) \tag{47}$$

s.t. (9), (13), (35) and

$$\sum_{t\in T}\sum_{k\in K}\left(\sum_{i\in N} 2z_{ikt}\varphi_{ikt} + \psi_{kt}(\boldsymbol{\vartheta}, \boldsymbol{\mu}, \boldsymbol{\mu^0})\right) \leq \tau \qquad \forall(\boldsymbol{\varphi}, \boldsymbol{\vartheta}, \boldsymbol{\mu}, \boldsymbol{\mu^0}) \in P_{\boldsymbol{\Theta}}. \tag{48}$$

This problem with the valid inequalities (16), (18)–(19) will be referred to as BMP2. Note that BMP2 is equivalent to a relaxation of the original problem (1)–(19) in which integrality is not imposed on the $x_{ijkt}$ variables. In Section 4.2, we explain how this relaxation can be used within an enumeration algorithm to obtain an optimal integer solution to the original problem.

Both BR1 and BR2 can be easily modified to handle the two-stage SIRP as follows. First, all the production setup variables $y_t$ in the BMP1 and BMP2 must be set to one, i.e., $y_t = \bar{y}_t = 1$. Second, the production quantity variables $p_{t\omega}$ and the parameters $M'_{it\omega}$ are substituted by $p_{t\omega} = B_t\bar{y}_t, \forall t \in T, \forall \omega \in \Omega$ and $\min\{L_i, Q\}$, respectively. Finally, constraints (25) and the dual variables $\delta_{t\omega}$ are removed from the PFS and DFS, respectively. The rest of the formulation remains unchanged.

# 4 Benders Decomposition Algorithms

In this section, we describe the algorithms we have developed to handle the Benders reformulations BR1 and BR2. We first present the classical algorithm and then introduce a Benders decomposition algorithm that is embedded in a branch-and-cut framework to solve the SPRP.

## 4.1 Classical Benders Decomposition Algorithm

In classical Benders decomposition, the master problem is solved to optimality at each iteration to obtain a lower bound on the optimal objective function value of the problem. Then, the subproblem is solved to generate a Benders cut and compute an upper bound. This cut is added to the master problem and the process is repeated until an optimal solution is found. In BR1, since the master problem BMP1 contains the SECs, it must be solved by a branch-and-cut process. At each node of the branch-and-bound tree, a so-called *separation algorithm* is applied to detect violated SECs and these cuts are then added to the problem. As in the algorithm of Adulyasak et al. (2012a), we use the minimum $s-t$ algorithm of the Concorde callable library (Applegate et al. 2011) as the exact separation algorithm. For BR2, since the routing part is handled by the PRS with the relaxed integrality constraints on the arc variables, the solution can be fractional. In this case, even when the $y_t$ and $z_{ikt}$ variables in the BMP2 take integer values, the value of $f\bar{y}_t$ plus the cost of the subproblems does not provide a valid upper bound on the original problem (1)–(19) because of the fractional $x_{ijkt}$ variables. However, an upper bound can be computed by solving the integer PRS either exactly or with a heuristic. We use the following heuristic to compute a valid upper bound when a fractional solution is obtained. TSP tours are constructed by the *GENIUS* procedure (Gendreau et al. 1992) and improved by *3-opt* routes (Lin 1965). This process is adequate since our experiments have shown that fractional solutions rarely occur and the gap between the heuristic solution and the optimal solution to the integer PRS is usually negligible because the number of customers in each tour is small. Note that even if we solve the integer PRS to optimality, a gap may still exist between the lower bound of the BMP2 and the best upper bound found by solving the integer PRS. If one wishes to guarantee an optimal solution using BR2, one can instead embed the Benders decomposition algorithm in a branch-and-cut algorithm as we will discuss in detail in the next section.

Denote by $P'$ the set of extreme points obtained so far by solving the Benders subproblems, i.e., DFS for BMP1, and DFS and PRS for BMP2. Note that BMP is used to represent the master problem (both

BMP1 and BMP2). We define $Z^b$ as the set of integer and binary variable values obtained by solving the BMP (consisting of $x, y, z$ for BMP1 and $y, z$ for BMP2). Let $BMP(P')$ be the Benders master problem with the Benders cuts generated from $P'$ and $SP(Z^b)$ be the Benders subproblems given the set of fixed integer and binary variables $Z^b$. Let also $v(BMP(P'))$, $v(SP(Z^b))$ and $\xi(Z^b)$ denote the objective function value of the Benders master problem, the objective function value of the Benders subproblems, and the cost associated with the fixed variables in the set $Z^b$, respectively. We further define $\tilde{v}(SP(Z^b))$ as the objective function value of the heuristic solution if the solution of the PRS is fractional in BR2, while we simply set $\tilde{v}(SP(Z^b)) = v(SP(Z^b))$ for BR1 and also for BR2 if the solution of the PRS is not fractional. Note that the algorithm stops when an optimal solution is found or a maximum CPU time is reached. The original Benders decomposition (BD) algorithm is shown in Algorithm 1.

**Algorithm 1** Original Benders Decomposition (BD) Algorithm

$ub, gap \leftarrow \infty, lb \leftarrow -\infty$ and $b \leftarrow 0$
$P' \leftarrow \varnothing$
**while** stopping criterion not satisfied **do**
  Solve $BMP(P')$ to obtain $Z^b$
  **if** $v(BMP(P')) > lb$ **then** $lb \leftarrow v(BMP(P'))$
  Solve $SP(Z^b)$ to obtain extreme points, generate Benders cuts and add to $BMP(P')$
  (for BMP2) Solve the TSP heuristic if $x$ is fractional
  **if** $\xi(Z^b) + \tilde{v}(SP(Z^b)) < ub$ **then** $ub \leftarrow \xi(Z^b) + \tilde{v}(SP(Z^b))$
  $gap \leftarrow (ub - lb)/ub$
  $b \leftarrow b + 1$
**end while**

When applying a Benders decomposition technique to a mixed-integer problem, one typically retains the integer variables in the BMP and solves the master problem from scratch by a branch-and-bound procedure at each iteration. This process may be highly inefficient. Furthermore, for BR1 where the master problem also contains the SECs, a branch-and-cut approach must be employed and the problem can be much more difficult to solve. To overcome these difficulties, we propose an alternative implementation of the Benders decomposition.

## 4.2 Branch-and-Cut Based Benders Algorithm

Since Benders cuts generated from any solution of the master problem (including the linear relaxation) are valid (McDaniel and Devine 1977), one can generate Benders cuts at any node of the branch-and-bound tree of the Benders master problem. This way, Benders decomposition can be embedded in a standard branch-and-cut framework where the Benders subproblems are solved and the generated Benders cuts are added to the master problem at any node of the branch-and-bound tree of the master problem. We refer to this approach as branch-and-Benders-cut (BBC). Other implementations of Benders decomposition in a branch-and-cut framework were discussed by Codato and Fischetti (2006) and Fortz and Poss (2009) to deal with feasibility cuts, as well as by Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) who employed an interior point method and an outer-approximation method to solve the Benders reformulations, respectively. In our algorithm, to avoid generating a large number of Benders cuts, the cuts are added to the branch-and-bound tree of the master problem only when an incumbent solution of the BMP is found. The process is described in Algorithm 2. A comparison of the cut generation strategy of each algorithm is shown in Table 1. It describes the stage at which each type of cuts is generated during the branch-and-bound process when solving the Benders master problem. Note that the Benders flow cuts (F) are generated by solving the DFS and the Benders route cuts (R) are generated by solving the PRS in our implementation.

As shown in Table 1, in BD, the Benders cuts are only generated and added when an optimal solution of the Benders master problem is found and the master problem is then solved from scratch with a new branch-and-bound tree in the next iteration. We thus explore $b$ branch-and-bound trees for the master problem where $b$ is the number of iterations in the Benders algorithm. In the BBC, however, a single branch-and-bound tree for the master problem is explored during the whole process. The SECs are added at each node.

**Algorithm 2** Branch-and-Benders-Cut (BBC) Algorithm

$ub, gap \leftarrow \infty, lb \leftarrow -\infty$ and $b \leftarrow 0$
$P' \leftarrow \varnothing$
**begin** solving $BMP(P')$ by a branch-and-bound and apply the following steps at each node of the branch-and-bound tree
  $lb \leftarrow$ the best overall lower bound of $BMP(P')$
  (for BMP1) Solve the separation algorithm and add SECs to $BMP(P')$
  **if** stopping criterion not satisfied **and** an incumbent solution found **then**
    Set $Z^b$ using the incumbent solution
    Solve $SP(Z^b)$ to obtain extreme points, generate Benders cuts and add to $BMP(P')$
    (for BMP2) Solve the TSP heuristic if $x$ is fractional
    **if** $\xi(Z^b) + \tilde{v}(SP(Z^b)) < ub$ **then** $ub \leftarrow \xi(Z^b) + \tilde{v}(SP(Z^b))$
    $gap \leftarrow (ub - lb)/ub$
    $b \leftarrow b + 1$
  **end if**
**end**

Table 1: Benders cut generation strategies of the Benders algorithms

| **Stage** | **Benders Reformulation** | | | |
|---|---|---|---|---|
| | **BR1** | | **BR2** | |
| | **BD** | **BBC** | **BD** | **BBC** |
| Any B&B Node | S | S | - | - |
| Incumbent | - | F, S | - | F,R |
| Optimal | F | n/a | F,R | n/a |
| **No. of B&B Trees** | $b$ | 1 | $b$ | 1 |

S - SECs
F - Benders flow cuts
R - Benders route cuts
$b$ - Number of iterations in Benders decomposition algorithm

Each time a new incumbent solution is found, the subproblems resulting from this incumbent solution are solved and Benders cuts are added. The branch-and-bound process continues until it reaches a stopping criterion. Both upper and lower bounds are computed at the nodes of the tree and an optimal solution to the original formulation (1)–(19) is found when the BMP has been solved to optimality. This process follows the branch-and-cut framework.

The fact that one tree is explored in the BBC algorithm is also particularly useful for BR2 when an integrality gap on the routing part exists and one must find an optimal solution for the original problem. This can be done by adding a few steps to the BBC. During the solution process, when a fractional PRS solution is found, the set $Z^b$ and the value $\xi(Z^b) + v(SP(Z^b))$ must be stored. This value is a lower bound on the solution cost corresponding to the set $Z^b$ for the original formulation and therefore the solution can be pruned later if this value exceeds the value of the current best overall feasible solution ($ub$). At the end of the process, one has a current best solution for the original problem and a (possibly empty) set of fractional solutions including the incumbent solution of the BMP2 if it is fractional. Then, all the remaining fractional solutions must be evaluated again by solving to optimality the PRS with the imposed integrality constraints (14) and (15). The optimal solution is then the best solution among all these solutions.

We also remark that this Benders cut generation strategy is different from those of Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) since the Benders cuts are added only when an incumbent solution of the BMP is found in our BBC algorithm, while Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) generate cuts at each node of the tree using a relaxed BMP solution. The latter strategy, however, appears to be inefficient for our problem as we show in the computational results reported in Section 5.1.

## 4.3 Computational Enhancements

### 4.3.1 Lower Bound Lifting Inequalities

Since parts of the objective function (1) are projected out in the Benders reformulations, the optimality gap may be large in the initial stage of the algorithm due to the low quality of the lower bound. A large number of Benders cuts are thus needed to close the gap. To address this issue, we can lift the lower bound of the Benders master problem by using initial cuts, called lower bound lifting inequalities (LBL), that contain some information about the parts of the original objective function that are removed. First, we can add cuts to represent a lower bound on the flow costs, i.e., unit production, inventory and penalty costs. We observe that, for the periods between two consecutive deliveries to a customer, the minimum flow cost can be calculated by considering the quantity that must be supplied between the two visits to satisfy the demand. Figure 1 illustrates the inventory level when customer $i$ is visited in period 2 and then in period 6 while the demand in each period is equal to 40 so that the minimum total quantity required to satisfy the demand between these periods is equal to 160. Given the maximum inventory capacity $L_i = 100$, this total demand quantity can be separated into two parts, i.e., the amount $\overline{A} = 100$ under $L_i$ which can be supplied and can be also left unmet if the cost of this is lower, and the amount $\overline{B} = 60$ which cannot be supplied and has to be left unmet due to the inventory capacity limit.

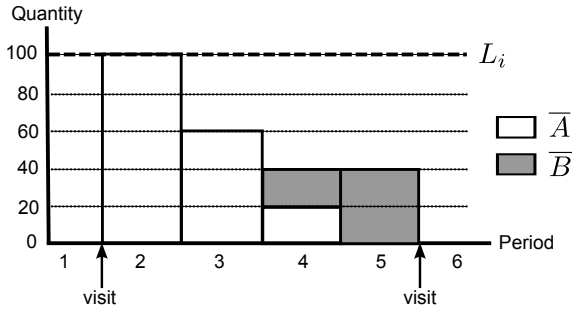


Figure 1: Inventory level corresponding to the two consecutive visits in periods 2 and 6

We define the periods 0 and $l+1$ as dummy periods at the beginning and the end of the planning horizon (used for calculation purposes) and $d_{i0\omega} = d_{i,l+1,\omega} = 0, \forall i \in N_c, \forall \omega \in \Omega$. Let $\lambda_{ivt}$ be a binary variable equal to one if customer $i$ is visited in period $v < t$ and the next visit is in period $t$, and the parameter $\phi_{ivt}$ be the minimum possible sum of unit production, inventory and penalty costs over the period $v$ to $t-1$ associated with the variable $\lambda_{ivt}$, calculated as,

$$\phi_{ivt} = \sum_{\omega \in \Omega} \rho_\omega \left( \sum_{s=v}^{t-1} \min \left\{ H^P_{ivs\omega}, H^\sigma_{ivs\omega} \right\} + c^h_{ivt} + \sigma_i \left( \sum_{s=v}^{t-1} d_{is\omega} - \varphi_{iv} \right)^+ \right)$$

where

$$H^P_{ivs\omega} = \begin{cases} (h_i(s-1)+u) \min \left\{ d_{is\omega}, \left( \varphi_{i0} - \sum_{w=1}^{s-1} d_{iw\omega} \right)^+ \right\} & \text{if } v = 0 \\ (h_i(s-v)+u) \min \left\{ d_{is\omega}, \left( \varphi_{iv} - \sum_{w=v}^{s-1} d_{iw\omega} \right)^+ \right\} & \text{if } v > 0, \end{cases}$$

$$H^\sigma_{ivs\omega} = \sigma_i \min \left\{ d_{is\omega}, \left( \varphi_{iv} - \sum_{w=v}^{s-1} d_{iw\omega} \right)^+ \right\},$$

$$\varphi_{iv} = \begin{cases} I_{i0} & \text{if } v = 0 \\ L_i & \text{if } 0 < v < t \leq l+1 \end{cases} \text{ and } c^h_{ivt} = \begin{cases} h_i(t-1) \left( I_{i0} - \sum_{w=1}^{t-1} d_{iw\omega} \right)^+ & \text{if } v = 0 \\ h_i(t-v) \left( I_{i0} - \sum_{w=1}^{t-1} d_{iw\omega} \right)^+ & \text{if } 0 < v < t \leq l+1. \end{cases}$$

Note that $c^h_{ivt}$ is the inventory cost at customer $i$ incurred over the period $v$ to $t-1$ for the part of the initial inventory that is not used up at the end of period $t-1$. The following cuts can be added to the master problem:

$$\sum_{t=1}^{l+1}\sum_{v=0}^{t-1}\sum_{i\in N_c}\phi_{ivt}\lambda_{ivt} - u\sum_{i\in N_c} I_{i0} \leq \eta \tag{49}$$

$$\sum_{v=0}^{t-1}\lambda_{ivt} = \sum_{k\in K} z_{ikt} \qquad \forall i \in N_c, \forall t \in T \tag{50}$$

$$\sum_{v=0}^{t-1}\sum_{s=t}^{l+1}\lambda_{ivs} = 1 \qquad \forall i \in N_c, \forall t \in T \cup \{l+1\}. \tag{51}$$

Constraint (49) provides a lower bound for the flow cost. Constraints (50) link the $\lambda_{ivt}$ and $z_{ikt}$ variables and constraints (51) enforce that one replenishment plan $\lambda_{ivt}$ be selected in each period. Although the new variables $\lambda_{ivt}$ are added to the formulation, one can observe that when all the $z_{ikt}$ variables are fixed, the variables $\lambda_{ivt}$ can be easily set by inspection. These constraints can be added to both formulations BMP1 and BMP2.

For BR2, where the routing cost is also projected out, we let $\chi_{ijkt}$ be a routing variable which is identical to $x_{ijkt}$. The following inequalities can be added to the BMP2:

$$\sum_{(i,j)\in E}\sum_{k\in K} c_{ij}\chi_{ijkt} \leq \tau \tag{52}$$

$$\sum_{(j,j')\in\delta(i)} \chi_{jj'kt} = 2z_{ikt} \qquad \forall i \in N, \forall k \in K, \forall t \in T. \tag{53}$$

### 4.3.2 Scenario Group Cuts

In BR1 and BR2, the Benders subproblem is decomposed into many subproblems. Many Benders cuts, one for each subproblem, can be added at once instead of just one Benders cut to accelerate the convergence (Birge and Louveaux 1988). However, adding too many cuts at each iteration can lead to a decreased performance of the Benders algorithm (de Camargo et al. 2008). In the SPRP, where the number of subproblems is equal to the number of scenarios, the number of Benders cuts generated at each iteration can be very large. To overcome this issue, we can instead create groups of scenarios using some similarity and aggregate the Benders cuts in each group to reduce the number of cuts added per iteration. To create scenario groups, we first define the number of scenario groups $n_g \leq |\Omega|$ and groups of scenarios $G(g)$, indexed by $g$. Then, the range between the maximum and minimum total demand among all scenarios is calculated and separated equally into $n_g$ groups. For each scenario, if the total demand falls into the total demand range of a group, the scenario is then assigned to this group. Denote by $\eta_g$ the expected total flow cost corresponding to group $g$. The variable $\eta$ in the objective function (34) and (47) is replaced by $\sum_{g=1}^{n_g}\eta_g$ and constraints (35) are replaced by the following constraints:

$$\sum_{\omega\in G(g)}\rho_\omega\left(-\sum_{t\in T}M_{t\omega}\delta_{t\omega}y_t - \sum_{t\in T}\sum_{k\in K}Q\kappa_{kt\omega}z_{0kt} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c}M'_{it\omega}\zeta_{ikt\omega}z_{ikt} + \pi_\omega(\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\theta})\right) \leq \eta_g$$
$$\forall 1 \leq g \leq n_g, \forall(\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\theta},\boldsymbol{\delta},\boldsymbol{\zeta},\boldsymbol{\kappa}) \in P_\Delta. \tag{54}$$

In this way, we can control the number of Benders cuts added at each iteration. For the formulation BMP2, where the routing subproblems have to be solved, we can also add the cut for each vehicle in each period separately. Let $\tau_{kt}$ be the routing cost of vehicle $k$ in period $t$. The variable $\tau$ in the objective function (47) is replaced with $\sum_{t\in T}\sum_{k\in K}\tau_{kt}$ and constraints (48) are replaced with

$$\left(\sum_{i\in N}2z_{ikt}\varphi_{ikt} + \psi_{kt}(\boldsymbol{\vartheta},\boldsymbol{\mu},\boldsymbol{\mu^0})\right) \leq \tau_{kt} \qquad \forall k \in K, \forall t \in T, \forall(\boldsymbol{\varphi},\boldsymbol{\vartheta},\boldsymbol{\mu},\boldsymbol{\mu^0}) \in P_{\boldsymbol{\Theta}}. \tag{55}$$

### 4.3.3 Pareto-Optimal Cuts

The performance of a Benders decomposition approach depends largely on the quality of the cuts. Magnanti and Wong (1981) introduced the concept of non-dominated cuts, called *Pareto-optimal cuts*. Let $\mathbf{Y}$ and $\mathbf{Z}$ be the sets of vectors associated with the $y_t$ and $z_{ikt}$ variables, respectively. For the flow cut obtained by the DFS corresponding to the scenario $\omega$, the cut generated from the extreme point $(\boldsymbol{\alpha}^1, \boldsymbol{\beta}^1, \boldsymbol{\gamma}^1, \boldsymbol{\theta}^1, \boldsymbol{\delta}^1, \boldsymbol{\zeta}^1, \boldsymbol{\kappa}^1)$ dominates the cut generated from the extreme point $(\boldsymbol{\alpha}^2, \boldsymbol{\beta}^2, \boldsymbol{\gamma}^2, \boldsymbol{\theta}^2, \boldsymbol{\delta}^2, \boldsymbol{\zeta}^2, \boldsymbol{\kappa}^2)$ if and only if

$$
\begin{aligned}
-\sum_{t\in T} M_{t\omega}\delta^1_{t\omega} y_t - \sum_{t\in T}\sum_{k\in K} Q\kappa^1_{kt\omega} z_{0kt} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} M'_{it\omega}\zeta^1_{ikt\omega} z_{ikt} + \pi_\omega(\boldsymbol{\alpha}^1, \boldsymbol{\beta}^1, \boldsymbol{\gamma}^1, \boldsymbol{\theta}^1) \geq \\
-\sum_{t\in T} M_{t\omega}\delta^2_{t\omega} y_t - \sum_{t\in T}\sum_{k\in K} Q\kappa^2_{kt\omega} z_{0kt} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} M'_{it\omega}\zeta^2_{ikt\omega} z_{ikt} + \pi_\omega(\boldsymbol{\alpha}^2, \boldsymbol{\beta}^2, \boldsymbol{\gamma}^2, \boldsymbol{\theta}^2) \\
\forall \mathbf{y} \in \mathbf{Y}, \forall \mathbf{z} \in \mathbf{Z},
\end{aligned}
$$

with strict inequality for at least one point. Denote by $\mathbf{Y}^{LP}$ the polyhedron defined by $0 \leq y_t \leq 1, \forall t \in T$ and $\mathbf{Z}^{LP}$ the polyhedron defined by constraints (9) and $0 \leq z_{ikt} \leq 1, \forall i \in N, \forall k \in K, \forall t \in T$. Let $ri(\mathbf{Y}^{LP})$ and $ri(\mathbf{Z}^{LP})$ be the relative interior of $\mathbf{Y}^{LP}$ and $\mathbf{Z}^{LP}$, respectively. A Pareto-optimal cut can be obtained by solving the following subproblem, where $\mathbf{y}^0 \in ri(\mathbf{Y}^{LP})$ and $\mathbf{z}^0 \in ri(\mathbf{Z}^{LP})$:

$$
\begin{aligned}
\max -I_{00}\alpha_{1\omega} + \sum_{i\in N_c}(d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i\in N_c}\sum_{t=2}^{l} d_{it\omega}\beta_{it} - \sum_{t\in T} L_0\gamma_{t\omega} - \sum_{t\in T}\sum_{i\in N_c}(L_i - d_{it})\theta_{it\omega} \\
- \sum_{t\in T} M_{t\omega} y^0_t \delta_{t\omega} - \sum_{t\in T}\sum_{k\in K} Q z^0_{0kt}\kappa_{kt\omega} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} \overline{M}_{it\omega} z^0_{ikt}\zeta_{ikt\omega} \qquad (56)
\end{aligned}
$$

$$
\begin{aligned}
\text{s.t.} \quad -I_{00}\alpha_{1\omega} + \sum_{i\in N_c}(d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i\in N_c}\sum_{t=2}^{l} d_{it\omega}\beta_{it} - \sum_{t\in T} L_0\gamma_{t\omega} - \sum_{t\in T}\sum_{i\in N_c}(L_i - d_{it})\theta_{it\omega} \\
- \sum_{t\in T} M_{t\omega}\bar{y}_t\delta_{t\omega} - \sum_{t\in T}\sum_{k\in K} Q\bar{z}_{0kt}\kappa_{kt\omega} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} \overline{M}_{it\omega}\bar{z}_{ikt}\zeta_{ikt\omega} = \upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}}) \qquad (57) \\
(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}, \boldsymbol{\delta}, \boldsymbol{\zeta}, \boldsymbol{\kappa}) \in \Delta_\omega. \qquad (58)
\end{aligned}
$$

Constraints (57) and (58) ensure that we select a feasible dual solution that was optimal for the original DFS objective function $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$. To generate a Pareto-optimal cut, any core point $(y^0, z^0)$, where $\mathbf{y}^0 \in ri(\mathbf{Y}^{LP})$ and $\mathbf{z}^0 \in ri(\mathbf{Z}^{LP})$, can be used (Magnanti and Wong 1981). We employ a method similar to that of Cordeau et al. (2001) to ensure that the core point lies within the relative interior of the master problem polyhedron. The details of this method are provided in the Appendix.

The generation of Pareto-optimal cuts usually improves the convergence of the algorithm but requires solving two different linear programs sequentially for each subproblem, i.e., first the DFS to find $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$, and then the problem (56)–(58). We expedite the process by using a network flow algorithm to determine the value $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$.

## 5 Computational Experiments

We have performed experiments using the PRP instances introduced by Adulyasak et al. (2012a), which were themselves generated from the instances of Archetti et al. (2011). The test set consists of instances with $n$ = 10, 15, 20, 25 and 30 customers and $l$ = 3 and 6 periods. There are four instances per instance size and each of them has different characteristics. More details on the instances are provided in the Appendix. We generate scenarios by a Monte-Carlo simulation in which the demand in each period is independent and varies in the range $[\bar{d}_{it}(1-\epsilon), \bar{d}_{it}(1+\epsilon)]$, where $\bar{d}_{it}$ is the demand of the nominal case from the original test set. Unless stated otherwise, we assume that the demands are uniformly distributed and the value of $\epsilon$ is set to 0.1 and 0.25, which represents a maximum demand variation of 10% and 25%, respectively. The sets $\mathcal{S}$

and $\mathcal{L}$ are used to represent the instance sizes $n = 10$, 15 and $n = 20$, 25, 30, respectively. Columns *Gap*, *CPU* and *B.Cuts* show the average optimality gap (%), the average CPU time in seconds and the average number of generated Benders cuts, respectively. To indicate the best results, boldface letters are put on the smallest average computing time if all instances of a problem size are solved to optimality, otherwise they are put on the best average optimality gap. The experiments were conducted on a workstation with an Intel Xeon 2.67GHz processor and 24GB of RAM under Scientific Linux 6.1 using CPLEX 12.3. The algorithms were coded in C and C# on MonoDevelop 3.0 under OpenSUSE Linux 12.1. In all experiments, branching priority was given first to $y$ variables and then to $z$ and $x$ (for BR1) variables, respectively.

## 5.1 Performance of the Branch-and-Benders-Cut

This section explores the performance of the BBC algorithm and the effect of the computational enhancements. We chose the test set with $l = 3$ for these experiments. The number of vehicles is set to one and the number of scenarios is set to 100 and 500. The maximum computing time is set to two hours for each instance.

### 5.1.1 Comparison Between the Original Benders Algorithm and the Branch-and-Benders-Cut Algorithm

We first examine the performance of the BD and BBC algorithms for BR1 and BR2. For the BBC, we also explore the performance of the following different cut generation strategies:

***Every:*** Benders cuts are generated at every node in the branch-and-bound tree;

***Root/Incumbent:*** Benders cuts are only generated at the root node and when an incumbent solution of the BMP is found;

***Incumbent:*** Benders cuts are generated only when an incumbent solution of the BMP is found.

Note that the *Every* strategy was used by Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) while the *Incumbent* is the strategy we proposed in Algorithm 2. The *Root/Incumbent* can be seen as a combination of the previous two strategies. Tables 2 and 3 provide the average results for each instance set for BR1 and BR2, respectively.

Among the three Benders cut generation strategies for the BBC algorithm, the *Every* strategy is the worst performing and is even worse than the BD algorithm. The *Root/Incumbent* and *Incumbent* strategies have

Table 2: Comparison of the BD and BBC algorithms on BR1

| Set | $\|\Omega\|$ | $\epsilon$ | BD | | | | BBC | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | *Every* | | | | *Root/Incumbent* | | | | *Incumbent* | | | |
| | | | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | 1555.4 | 112.3 | 7/8 | 0.0 | 1748.6 | 9995.0 | 8/8 | 0.0 | **110.8** | 263.5 | 8/8 | 0.0 | 114.5 | 252.3 |
| | | 0.25 | 7/8 | 0.0 | 3062.3 | 178.6 | 7/8 | 0.0 | 1879.8 | 10642.9 | 8/8 | 0.0 | **106.3** | 360.4 | 8/8 | 0.0 | 203.5 | 328.5 |
| | 500 | 0.10 | 8/8 | 0.0 | 1524.4 | 114.0 | 6/8 | 0.8 | 3189.8 | 3900.9 | 8/8 | 0.0 | **225.7** | 271.9 | 8/8 | 0.0 | 302.4 | 250.3 |
| | | 0.25 | 7/8 | 0.2 | 2816.4 | 144.4 | 4/8 | 1.6 | 3901.8 | 5015.9 | 8/8 | 0.0 | 316.8 | 324.5 | 8/8 | 0.0 | **287.8** | 268.6 |
| Average | | | 30/32 | 0.0 | 2239.6 | 137.3 | 24/32 | 0.6 | 2680.0 | 7388.7 | 32/32 | 0.0 | **189.9** | 305.1 | 32/32 | 0.0 | 227.1 | 274.9 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 7.4 | 7200.0 | 79.7 | 0/12 | 5.8 | 4800.0 | 20853.8 | 0/12 | 4.9 | 7200.0 | 3392.3 | 0/12 | **4.7** | 7200.0 | 2181.7 |
| | | 0.25 | 0/12 | 7.3 | 7200.0 | 83.4 | 0/12 | 7.7 | 7200.0 | 34209.6 | 0/12 | 5.7 | 7200.0 | 4022.5 | 0/12 | **4.8** | 7200.0 | 2435.7 |
| | 500 | 0.10 | 0/12 | 7.6 | 7200.0 | 76.7 | 0/12 | 9.9 | 7200.0 | 10267.9 | 1/12 | **4.8** | 7131.8 | 2916.0 | 0/12 | 5.1 | 7200.0 | 2001.0 |
| | | 0.25 | 0/12 | 8.7 | 7200.0 | 88.3 | 0/12 | 10.6 | 7200.0 | 10240.8 | 0/12 | 6.0 | 7200.0 | 3360.1 | 0/12 | **5.2** | 7200.0 | 2371.2 |
| Average | | | 0/48 | 7.7 | 7200.0 | 82.0 | 0/48 | 8.5 | 6600.0 | 18893.0 | 1/48 | 5.3 | 7183.0 | 3422.7 | 0/48 | **4.9** | 7200.0 | 2247.4 |
| Total | | | 30/80 | 4.7 | 5215.8 | 104.1 | 24/80 | 5.4 | 5032.0 | 14291.3 | 33/80 | 3.2 | 4385.7 | 2175.7 | 32/80 | **3.0** | 4410.8 | 1458.4 |

Table 3: Comparison of the BD and BBC algorithms on BR2

| Set | $\|\Omega\|$ | $\epsilon$ | BD | | | | BBC | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Every | | | | Root/Incumbent | | | | Incumbent | | | |
| | | | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 2/8 | 7.3 | 5981.4 | 575.0 | 3/8 | 7.3 | 5271.4 | 42714.3 | 4/8 | 5.1 | 3994.3 | 9328.3 | 4/8 | **4.5** | 3906.0 | 8967.8 |
| | | 0.25 | 1/8 | 7.2 | 6348.0 | 607.8 | 4/8 | 7.3 | 5441.7 | 45463.0 | 4/8 | 5.7 | 3798.5 | 10305.8 | 4/8 | **5.1** | 3731.0 | 8429.3 |
| | 500 | 0.10 | 2/8 | 6.9 | 6117.7 | 585.5 | 1/8 | 11.9 | 6639.7 | 17163.8 | 4/8 | **4.8** | 4105.7 | 6601.0 | 4/8 | 5.2 | 4089.8 | 5806.8 |
| | | 0.25 | 1/8 | 8.0 | 6370.7 | 608.5 | 1/8 | 10.7 | 6840.4 | 17860.0 | 4/8 | **5.5** | 4186.5 | 7321.5 | 4/8 | 6.2 | 4052.9 | 7102.3 |
| Average | | | 6/32 | 7.3 | 6204.5 | 594.2 | 9/32 | 9.3 | 6048.3 | 30800.3 | 16/32 | **5.3** | 4021.2 | 8389.1 | 16/32 | **5.3** | 3944.9 | 7576.5 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 25.0 | 7200.0 | 627.7 | 0/12 | 25.2 | 7200.0 | 34952.8 | 0/12 | 17.3 | 7200.0 | 13226.0 | 0/12 | **16.3** | 7200.0 | 13859.5 |
| | | 0.25 | 0/12 | 26.9 | 7200.0 | 480.0 | 0/12 | 25.7 | 7200.0 | 34744.8 | 0/12 | 19.0 | 7200.0 | 14767.7 | 0/12 | **13.8** | 7200.0 | 11261.7 |
| | 500 | 0.10 | 0/12 | 25.3 | 7200.0 | 540.2 | 0/12 | 32.7 | 7200.0 | 11935.7 | 0/12 | **17.8** | 7200.0 | 8491.5 | 0/12 | 18.4 | 7200.0 | 8573.5 |
| | | 0.25 | 0/12 | 25.9 | 7200.0 | 692.7 | 0/12 | 34.4 | 7200.0 | 11605.5 | 0/12 | 19.0 | 7200.0 | 8235.3 | 0/12 | **18.8** | 7200.0 | 8312.2 |
| Average | | | 0/48 | 25.8 | 7200.0 | 585.1 | 0/48 | 29.5 | 7200.0 | 23309.7 | 0/48 | 18.3 | 7200.0 | 11180.1 | 0/48 | **16.8** | 7200.0 | 10501.7 |
| Total | | | 6/80 | 18.4 | 6801.8 | 588.8 | 9/80 | 21.4 | 6739.3 | 26305.9 | 16/80 | 13.1 | 5928.5 | 10063.7 | 16/80 | **12.2** | 5898.0 | 9331.6 |

similar performance but the *Incumbent* is better overall for both BR1 and BR2. The number of Benders cuts generated by the *Every* strategy is much higher than those generated by *Root/Incumbent* and *Incumbent*. When comparing the BBC with *Incumbent* and the BD algorithm, the results clearly indicate the benefits of using the branch-and-cut framework to solve the Benders reformulations. The average optimality gaps after two hours of CPU time as well as the computing times on the instances solved to optimality are improved considerably. It should also be noted that the average number of Benders cuts in the BBC is much larger than in the BD and this could strengthen the master problem and expedite the process. For these reasons, we use only the BBC with *Incumbent* for the remaining computational experiments. We also observe that the performance of BR2 is significantly worse than BR1 because the routing parts are removed from the master problem. However, since the methods presented in Section 4.3 can further improve the quality of the cuts and the process, we still perform the experiments on BR2 using the BBC to see the impact of the computational enhancements.

### 5.1.2 Impact of the Lower Bound Lifting Inequalities

Table 4 reports the performance of the BBC algorithm when the lower bound lifting cuts (LBL) of Section 4.3.1 are added. The results show that the LBL substantially improves the performance of the Benders algorithm. For the instance set $\mathcal{S}$ solved to optimality by BR1, the average computing time is reduced by 75%. The average optimality gap is also significantly decreased to approximately one third of the previous average gap and the BBC algorithm could solve an additional 16 and 13 instances to optimality for BR1 and BR2, respectively. The results clearly indicate the benefits of the LBL. These lower bound lifting inequalities are thus used in the remaining parts of the computational experiments.

Table 4: Average results of the BBC algorithms using the lower bound lifting inequalities (LBL)

| Set | \|Ω\| | ϵ | Benders Reformulation 1 (BR1) | | | | | | | | Benders Reformulation 2 (BR2) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | None | | | | LBL | | | | None | | | | LBL | | | |
| | | | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | 114.5 | 252.3 | 8/8 | 0.0 | **27.0** | 97.1 | 4/8 | 4.5 | 3906.0 | 8967.8 | 7/8 | **0.0** | 1361.3 | 3471.0 |
| | | 0.25 | 8/8 | 0.0 | 203.5 | 328.5 | 8/8 | 0.0 | **33.5** | 124.1 | 4/8 | 5.1 | 3731.0 | 8429.3 | 8/8 | 0.0 | **845.7** | 2467.5 |
| | 500 | 0.10 | 8/8 | 0.0 | 302.4 | 250.3 | 8/8 | 0.0 | **72.9** | 89.9 | 4/8 | 5.2 | 4089.8 | 5806.8 | 7/8 | **0.0** | 1855.9 | 3002.3 |
| | | 0.25 | 8/8 | 0.0 | 287.8 | 268.6 | 8/8 | 0.0 | **94.6** | 119.6 | 4/8 | 6.2 | 4052.9 | 7102.3 | 7/8 | **0.1** | 1515.7 | 2479.8 |
| Average | | | 32/32 | 0.0 | 227.1 | 274.9 | 32/32 | 0.0 | **57.0** | 107.7 | 16/32 | 5.3 | 3944.9 | 7576.5 | 29/32 | **0.0** | 1394.7 | 2855.1 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 4.7 | 7200.0 | 2181.7 | 4/12 | **1.7** | 5617.0 | 1762.8 | 0/12 | 16.3 | 7200.0 | 13859.5 | 0/12 | **5.4** | 7200.0 | 17612.2 |
| | | 0.25 | 0/12 | 4.8 | 7200.0 | 2435.7 | 4/12 | **1.7** | 5602.3 | 1904.9 | 0/12 | 13.8 | 7200.0 | 11261.7 | 0/12 | **5.9** | 7200.0 | 18055.5 |
| | 500 | 0.10 | 0/12 | 5.1 | 7200.0 | 2001.0 | 5/12 | **1.6** | 5523.3 | 1535.7 | 0/12 | 18.4 | 7200.0 | 8573.5 | 0/12 | **5.9** | 7200.0 | 9058.3 |
| | | 0.25 | 0/12 | 5.2 | 7200.0 | 2371.2 | 3/12 | **2.1** | 6044.8 | 1873.4 | 0/12 | 18.8 | 7200.0 | 8312.2 | 0/12 | **6.3** | 7200.0 | 8933.2 |
| Average | | | 0/48 | 4.9 | 7200.0 | 2247.4 | 16/48 | **1.8** | 5696.8 | 1769.2 | 0/48 | 16.8 | 7200.0 | 10501.7 | 0/48 | **5.9** | 7200.0 | 13414.8 |
| Total | | | 32/80 | 3.0 | 4410.8 | 1458.4 | 48/80 | **1.1** | 3440.9 | 1104.6 | 16/80 | 12.2 | 5898.0 | 9331.6 | 29/80 | **3.6** | 4877.9 | 9190.9 |

### 5.1.3 Impact of the Scenario Group Cuts and Pareto-Optimal Cuts

The average results of different settings are shown in Tables 5 and 6 for BR1 and BR2, respectively. Although we have tested several different choices for the number of groups in the scenario group cuts, we report results only for $b = 5$ and $b = |\Omega|$ because a change of $b$ in a small range (e.g., less than 10) has little impact on the performance. Note that the latter case (i.e., $b = |\Omega|$) is the same as the multicut strategy proposed by Birge and Louveaux (1988). The multiple route cuts (47)–(48) are also applied to BR2 when using the scenario group cuts. Table 7 provides details on the average number of nodes (in thousands), and average number of Benders cuts.

Table 5: Average results of the BBC algorithm using the scenario group cuts and Pareto-optimal cuts on BR1

| Set | \|Ω\| | ϵ | Single cut | | | | | | 5-Cut | | | | | | \|Ω\|-Cut | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | |
| | | | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | 27.0 | 8/8 | 0.0 | 50.9 | 8/8 | 0.0 | **25.4** | 8/8 | 0.0 | 38.4 | 8/8 | 0.0 | 33.5 | 8/8 | 0.0 | 35.1 |
| | | 0.25 | 8/8 | 0.0 | 33.5 | 8/8 | 0.0 | 46.9 | 8/8 | 0.0 | **23.8** | 8/8 | 0.0 | 41.3 | 8/8 | 0.0 | 40.3 | 8/8 | 0.0 | 51.5 |
| | 500 | 0.10 | 8/8 | 0.0 | **72.9** | 8/8 | 0.0 | 151.3 | 8/8 | 0.0 | 82.5 | 8/8 | 0.0 | 131.7 | 8/8 | 0.0 | 256.5 | 8/8 | 0.0 | 210.8 |
| | | 0.25 | 8/8 | 0.0 | 94.6 | 8/8 | 0.0 | 180.5 | 8/8 | 0.0 | **75.9** | 8/8 | 0.0 | 158.4 | 8/8 | 0.0 | 243.8 | 8/8 | 0.0 | 299.3 |
| Average | | | 32/32 | 0.0 | 57.0 | 32/32 | 0.0 | 107.4 | 32/32 | 0.0 | **51.9** | 32/32 | 0.0 | 92.5 | 32/32 | 0.0 | 143.5 | 32/32 | 0.0 | 149.2 |
| $\mathcal{L}$ | 100 | 0.10 | 4/12 | 1.7 | 5617.0 | 6/12 | 1.3 | 5218.9 | 3/12 | 1.9 | 5665.8 | 6/12 | **1.2** | 5176.2 | 2/12 | 2.8 | 6301.5 | 4/12 | 1.7 | 6030.0 |
| | | 0.25 | 4/12 | 1.7 | 5602.3 | 5/12 | 1.6 | 5473.5 | 4/12 | 2.0 | 5756.3 | 4/12 | **1.5** | 5239.1 | 3/12 | 2.4 | 6116.6 | 2/12 | 2.0 | 6125.3 |
| | 500 | 0.10 | 5/12 | **1.6** | 5523.3 | 4/12 | 1.7 | 6093.8 | 3/12 | 2.1 | 5781.4 | 5/12 | **1.6** | 5615.4 | 1/12 | 4.0 | 7049.1 | 2/12 | 3.4 | 6630.5 |
| | | 0.25 | 3/12 | 2.1 | 6044.8 | 2/12 | 2.4 | 6379.1 | 3/12 | 2.0 | 5927.1 | 3/12 | **1.9** | 6033.0 | 2/12 | 3.6 | 6971.1 | 2/12 | 3.3 | 6708.0 |
| Average | | | 16/48 | 1.8 | 5696.8 | 17/48 | 1.8 | 5791.3 | 13/48 | 2.0 | 5782.7 | 18/48 | **1.6** | 5515.9 | 8/48 | 3.2 | 6609.6 | 10/48 | 2.6 | 6373.5 |
| Total | | | 48/80 | 1.1 | 3440.9 | 49/80 | 1.1 | 3517.8 | 45/80 | 1.2 | 3490.4 | 50/80 | **0.9** | 3346.5 | 40/80 | 1.9 | 4023.2 | 42/80 | 1.6 | 3883.7 |

Table 6: Average results of the BBC algorithm using the scenario group cuts and Pareto-optimal cuts on BR2

| Set | \|Ω\| | ϵ | Single cut | | | | | | 5-Cut | | | | | | \|Ω\|-Cut | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | |
| | | | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU |
| $\mathcal{S}$ | 100 | 0.10 | 7/8 | 0.0 | 1361.3 | 7/8 | 0.0 | 1547.4 | 8/8 | 0.0 | 367.3 | 8/8 | 0.0 | **285.6** | 6/8 | 0.1 | 2196.1 | 6/8 | 0.1 | 1950.2 |
| | | 0.25 | 8/8 | 0.0 | 845.7 | 8/8 | 0.0 | 930.7 | 8/8 | 0.0 | 260.8 | 8/8 | 0.0 | **230.8** | 7/8 | 0.2 | 1599.7 | 7/8 | 0.2 | 1195.2 |
| | 500 | 0.10 | 7/8 | 0.0 | 1855.9 | 6/8 | 0.4 | 2320.4 | 8/8 | 0.0 | **506.5** | 8/8 | 0.0 | 776.5 | 4/8 | 1.0 | 3660.7 | 6/8 | 0.7 | 2859.3 |
| | | 0.25 | 7/8 | 0.1 | 1515.7 | 7/8 | 0.4 | 1922.0 | 8/8 | 0.0 | **386.1** | 8/8 | 0.0 | 705.6 | 4/8 | 1.2 | 3656.1 | 5/8 | 0.7 | 3380.3 |
| Average | | | 29/32 | 0.0 | 1394.7 | 28/32 | 0.2 | 1680.1 | 32/32 | 0.0 | **380.2** | 32/32 | 0.0 | 499.6 | 21/32 | 0.6 | 2778.1 | 24/32 | 0.4 | 2346.3 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 5.4 | 7200.0 | 0/12 | 5.4 | 7200.0 | 0/12 | 5.5 | 7200.0 | 0/12 | **5.0** | 7200.0 | 0/12 | 6.4 | 7200.0 | 0/12 | 6.1 | 7200.0 |
| | | 0.25 | 0/12 | 5.9 | 7200.0 | 0/12 | 5.6 | 7200.0 | 0/12 | 5.7 | 7200.0 | 0/12 | **5.3** | 7200.0 | 0/12 | 6.5 | 7200.0 | 0/12 | 6.0 | 7200.0 |
| | 500 | 0.10 | 0/12 | 5.9 | 7200.0 | 0/12 | 6.2 | 7200.0 | 0/12 | 5.8 | 7200.0 | 0/12 | **5.7** | 7200.0 | 0/12 | 7.7 | 7200.0 | 0/12 | 7.3 | 7200.0 |
| | | 0.25 | 0/12 | 6.3 | 7200.0 | 0/12 | 6.7 | 7200.0 | 0/12 | **5.9** | 7200.0 | 0/12 | 6.0 | 7200.0 | 0/12 | 7.6 | 7200.0 | 0/12 | 7.4 | 7200.0 |
| Average | | | 0/48 | 5.9 | 7200.0 | 0/48 | 6.0 | 7200.0 | 0/48 | 5.7 | 7200.0 | 0/48 | **5.5** | 7200.0 | 0/48 | 7.1 | 7200.0 | 0/48 | 6.7 | 7200.0 |
| Total | | | 29/80 | 3.6 | 4877.9 | 28/80 | 3.7 | 4992.1 | 32/80 | 3.4 | 4472.1 | 32/80 | **3.3** | 4519.9 | 21/80 | 4.5 | 5431.3 | 24/80 | 4.2 | 5258.5 |

According to the results reported in Tables 5 and 6, the performance of the $|\Omega|$-cut indicates that a large number of scenario group cuts can lead to a significant increase in computational effort because the problem size is too large to be solved efficiently. However, choosing an appropriate number of groups can lead to a better performance compared to a single cut as one can see in the results on the set $\mathcal{S}$ for both BR1 and BR2 and the set $\mathcal{L}$ for BR2. Pareto-optimal cuts can result in increased computing time on the small instance set $\mathcal{S}$ but the gap on the large instance set $\mathcal{L}$ generally decreases. Combining the 5-cut and Pareto-optimal cuts provides the best results compared to the other options. Table 7 clearly explains the impact of the scenario group cuts and Pareto-optimal cuts that lead to a faster convergence. The number of Benders cuts when using the $|\Omega|$-cut significantly increases compared to the single cut especially when the number of scenarios is large, while the number of Benders cuts grows to a much smaller extent when using the 5-cut. The Pareto-optimal cuts, however, could reduce the number of generated Benders cuts compared to the non-Pareto-optimal cuts on the instance set $\mathcal{S}$ solved to optimality and could provide smaller gaps on the remaining instances while fewer Benders cuts are generated. We also observe that, on the small instance set $\mathcal{S}$, the number of times in which the Benders subproblems are called to generate a set of Benders cuts associated with a BMP solution $(\bar{\mathbf{y}}, \bar{\mathbf{z}})$ decreased by 52% and 20% for BR1 and decreased by 90% and 80% for BR2 by using the $|\Omega|$-cut and 5-cut, respectively, compared to the single cut. We can also conclude that the performance of the BBC on BR1 is far superior to BR2. Thus, we choose the BBC with LBL, 5-cut and Pareto-optimal cut of BR1 as the preferred setting of the BBC algorithm for the remaining computational experiments.

Table 7: Average number of nodes (in thousands) and Benders cuts using the scenario group cuts and Pareto-optimal cuts

| | Set | Single cut | | | | 5-Cut | | | | \|Ω\|-Cut | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Non-Pareto | | Pareto | | Non-Pareto | | Pareto | | Non-Pareto | | Pareto | |
| | | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts |
| BR1 | $\mathcal{S}$ | 1.9 | 107.7 | 1.1 | 64.4 | 1.7 | 436.3 | 0.8 | 279.1 | 0.8 | 15584.4 | 6.0 | 12081.3 |
| | $\mathcal{L}$ | 81.9 | 1769.2 | 56.2 | 1080.3 | 55.1 | 7105.8 | 37.2 | 4280.9 | 10.5 | 90631.3 | 9.9 | 75758.3 |
| BR2 | $\mathcal{S}$ | 87.0 | 2855.1 | 52.9 | 1901.6 | 73.7 | 1929.4 | 43.8 | 1416.2 | 18.7 | 46649.6 | 18.2 | 42679.3 |
| | $\mathcal{L}$ | 144.3 | 13414.8 | 68.5 | 6657.5 | 201.1 | 23247.3 | 152.1 | 14052.1 | 22.9 | 131657.8 | 21.7 | 114537.4 |

## 5.2 Comparisons with the Branch-and-Cut Procedure (BC)

In this section, we compare the results of the BBC with a branch-and-cut algorithm (BC) similar to the approaches of Archetti et al. (2007, 2011), Solyalı and Süral (2011), Solyalı et al. (2012) and Adulyasak et al. (2012a) to solve several variants of the deterministic PRP and IRP. We adapted the branch-and-cut algorithm for the vehicle index formulation presented in Adulyasak et al. (2012a), which provided the best results for the deterministic problem, to deal with stochastic version. This BC algorithm is applied to solve the formulation $BF$. The results are shown in Table 8. Columns *N.Cplex* and *N.SECs* show the number of CPLEX cuts and SECs generated in the branch-and-bound tree, respectively. To better explore the differences between the two approaches, we performed the test with a number of scenarios $|\Omega| = 100$, 200, 500 and 1000. Note that the average optimality gap was computed only using instances where a feasible solution is found.

When the number of scenarios is not large ($|\Omega| = 100$ and 200), the BC algorithm can still solve the instances with $n = 30$ to optimality, while the BBC algorithm could not find the optimal solution for the

Table 8: Average results of the BC and the BBC algorithms

| Set | $\|\Omega\|$ | $\epsilon$ | BC | | | | | | BBC | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #Opt | Gap | CPU | Nodes | N.Cplex | N.SECs | #Opt | Gap | CPU | Nodes | N.Cplex | N.SECs | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | **27.1** | 32.4 | 2873.3 | 74.3 | 8/8 | 0.0 | 38.4 | 704.5 | 14.8 | 196.4 | 270.0 |
| | | 0.25 | 8/8 | 0.0 | **30.5** | 43.1 | 2966.4 | 77.5 | 8/8 | 0.0 | 41.3 | 712.0 | 13.3 | 211.0 | 308.1 |
| | 200 | 0.10 | 8/8 | 0.0 | 85.7 | 41.8 | 5569.5 | 76.0 | 8/8 | 0.0 | **60.7** | 849.4 | 18.8 | 209.6 | 250.6 |
| | | 0.25 | 8/8 | 0.0 | 88.4 | 42.8 | 6133.8 | 76.0 | 8/8 | 0.0 | **87.8** | 948.0 | 14.3 | 222.0 | 358.1 |
| | 500 | 0.10 | 8/8 | 0.0 | 499.8 | 53.1 | 13413.3 | 87.5 | 8/8 | 0.0 | **131.7** | 988.9 | 20.8 | 212.1 | 241.9 |
| | | 0.25 | 8/8 | 0.0 | 475.2 | 47.8 | 15143.4 | 91.9 | 8/8 | 0.0 | **158.4** | 773.8 | 17.0 | 195.8 | 296.3 |
| | 1000 | 0.10 | 8/8 | 0.0 | 2068.2 | 62.9 | 25796.9 | 93.5 | 8/8 | 0.0 | **281.3** | 700.4 | 18.1 | 202.6 | 292.5 |
| | | 0.25 | 8/8 | 0.0 | 1994.8 | 53.4 | 28207.3 | 91.1 | 8/8 | 0.0 | **357.6** | 856.3 | 17.1 | 237.9 | 366.9 |
| Average | | | 64/64 | 0.0 | 658.7 | 47.1 | 12513.0 | 83.5 | 64/64 | 0.0 | **144.7** | 816.6 | 16.8 | 210.9 | 298.0 |
| $\mathcal{L}$ | 100 | 0.10 | 12/12 | 0.0 | **669.8** | 425.8 | 6154.7 | 470.8 | 6/12 | 1.2 | 5176.2 | 47558.3 | 22.2 | 1910.2 | 4594.2 |
| | | 0.25 | 12/12 | 0.0 | **660.9** | 337.0 | 6603.4 | 455.9 | 4/12 | 1.5 | 5239.1 | 45597.8 | 22.0 | 1824.1 | 5073.3 |
| | 200 | 0.10 | 12/12 | 0.0 | **1706.9** | 322.2 | 13051.6 | 436.0 | 6/12 | 1.4 | 5237.7 | 40121.6 | 21.6 | 1885.9 | 4060.0 |
| | | 0.25 | 12/12 | 0.0 | **1844.0** | 386.1 | 13072.0 | 452.8 | 5/12 | 1.7 | 5513.1 | 36761.8 | 24.4 | 1838.3 | 5033.8 |
| | 500 | 0.10 | 7/12 | $0.4^{(3)}$ | 4957.6 | 161.1 | 30650.6 | 335.8 | 5/12 | **1.6** | 5615.4 | 28579.6 | 21.1 | 1714.2 | 3484.2 |
| | | 0.25 | 7/12 | $3.7^{(1)}$ | 5169.1 | 175.4 | 31821.2 | 332.5 | 3/12 | **1.9** | 6033.0 | 26865.2 | 21.6 | 1757.9 | 3972.1 |
| | 1000 | 0.10 | 2/12 | $12.8^{(6)}$ | 6918.1 | 27.8 | 58806.1 | 108.1 | 4/12 | **2.1** | 6185.7 | 22172.7 | 22.7 | 1626.4 | 2634.2 |
| | | 0.25 | 2/12 | $13.0^{(7)}$ | 6860.6 | 21.6 | 57049.8 | 90.6 | 3/12 | **2.5** | 6307.3 | 19517.9 | 21.1 | 1494.3 | 2763.8 |
| Average | | | 66/96 | $3.7^{(17)}$ | 3598.4 | 232.1 | 27151.2 | 335.3 | 36/96 | **1.8** | 5663.4 | 33396.9 | 22.1 | 1756.4 | 3951.9 |
| Total | | | 130/160 | $2.2^{(17)}$ | 2422.5 | 158.1 | 21295.9 | 234.6 | 100/160 | **1.1** | 3455.9 | 20364.8 | 19.9 | 1138.2 | 2490.4 |

$^{(-)}$ the number of instance where a feasible solution could not be found

large instances within the time limit. When the number of scenarios is larger ($|\Omega| = 500$ and 1000), however, the BC is inferior to the BBC and it could not even find a feasible solution for several instances. We observe that the BBC is far less sensitive to the number of scenarios.

In addition to this test, we solved similar instances with more vehicles ($m = 2$) and a longer planning horizon ($l = 6$) to see the impact of these parameters. The results are provided in Table 9. When the number of vehicles or time periods increases, the performance of the BC algorithm drops significantly. The BBC algorithm outperforms the BC algorithm on most of the instances except on those with $|\Omega| = 100$ and on the instance set $\mathcal{S}$ with 6 periods and $|\Omega| = 200$. The BC algorithm, however, cannot handle the instances with $|\Omega| \geq 500$ and could not find a feasible solution for many of these instances. For the set $\mathcal{L}$ with $|\Omega| = 1000$, the root node could not even be processed within two hours.

Table 9: Average results of the BC and the BBC algorithms on instances with more vehicles and a longer horizon

| Set | $\|\Omega\|$ | 2 vehicles ($m = 2$) | | | | | | | | 6 periods ($l = 6$) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | BC | | | | BBC | | | | BC | | | | BBC | | | |
| | | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes |
| $\mathcal{S}$ | 100 | 16/16 | 0.0 | 144.6 | 84.4 | 16/16 | 0.0 | **74.7** | 1193.1 | 15/16 | 0.0 | **1229.0** | 1029.7 | 10/16 | 0.6 | 3519.5 | 29655.0 |
| | 200 | 16/16 | 0.0 | 377.6 | 84.9 | 16/16 | 0.0 | **123.3** | 1286.6 | 14/16 | **0.2** | 2724.0 | 637.5 | 8/16 | 0.8 | 3843.4 | 23895.9 |
| | 500 | 15/16 | 0.5 | 2755.7 | 100.8 | 16/16 | 0.0 | **214.9** | 1276.1 | 6/16 | 7.1 | 5893.5 | 155.1 | 8/16 | **1.3** | 4129.0 | 11724.4 |
| | 1000 | 9/16 | $13.7^{(3)}$ | 4533.9 | 17.4 | 16/16 | 0.0 | **352.8** | 1261.2 | 0/16 | $19.0^{(14)}$ | 7200.0 | 4.5 | 8/16 | **1.5** | 4453.5 | 6582.5 |
| Average | | 56/64 | 3.5 | 1953.0 | 71.9 | 64/64 | 0.0 | **191.4** | 1254.2 | 35/64 | 6.6 | 4261.6 | 456.7 | 34/64 | **1.1** | 3986.3 | 17964.5 |
| $\mathcal{L}$ | 100 | 16/24 | **0.6** | 3834.4 | 697.2 | 10/24 | 1.6 | 5451.9 | 21881.2 | 4/24 | **3.5** | 6403.6 | 1085.8 | 0/24 | 7.1 | 7200.0 | 17084.2 |
| | 200 | 12/24 | 6.7 | 5368.2 | 286.6 | 8/24 | **1.9** | 5702.0 | 19840.0 | 3/24 | 13.3 | 6916.7 | 327.2 | 0/24 | **7.7** | 7200.0 | 12486.7 |
| | 500 | 1/24 | $23.3^{(13)}$ | 7198.8 | 33.9 | 5/24 | **2.0** | 6167.5 | 16510.9 | 0/24 | $32.1^{(18)}$ | 7200.0 | 1.0 | 0/24 | **8.6** | 7200.0 | 6501.3 |
| | 1000 | 0/24 | $n/a^{(24)}$ | 7200.0 | 0.0 | 6/24 | **2.3** | 6274.4 | 10856.0 | 0/24 | $n/a^{(24)}$ | 7200.0 | 0.0 | 0/24 | **9.7** | 7200.0 | 3721.6 |
| Average | | 29/96 | $10.2^{(37)}$ | 5900.4 | 254.4 | 29/96 | **1.9** | 5899.0 | 17272.0 | 7/96 | $16.3^{(42)}$ | 6930.1 | 353.5 | 0/96 | **8.3** | 7200.0 | 9948.4 |
| Total | | 85/160 | $7.5^{(40)}$ | 4321.4 | 181.4 | 93/160 | **1.2** | 3616.0 | 10864.9 | 42/160 | $12.4^{(42)}$ | 5862.7 | 394.8 | 34/160 | **5.4** | 5914.5 | 13154.8 |

$^{(-)}$ the number of instance where a feasible solution could not be found

It should also be noted that a large number of CPLEX cuts is generated when solving the BC and the majority of them, approximately 90%, are *flow cover* cuts that strengthen the network structure of the problem. In Table 10, we further examine the impact of turning off CPLEX cuts on the instances with $m = 1$ and $l = 3$. The results clearly show the impact of the CPLEX cuts for the BC algorithm. Without these cuts, the performance of the BC algorithm decreases significantly. The number of optimal solutions found by the BC algorithm after two hours on the instance set $\mathcal{L}$ decreased from 66 to 31 and the instances with $|\Omega| \leq 200$ were not solved to optimality. This can be explained by the substantial increase in the average number of nodes of the BC algorithm when the CPLEX cuts are not used. However, more feasible solutions on large instances could be found because the BC explored a larger number of nodes in the branch-and-bound tree. For the BBC, there is almost no impact on the performance and the algorithm could also provide better results on the instance set $\mathcal{L}$ with $|\Omega| = 200$ compared to the BC when the CPLEX cuts are turned-off.

Table 10: Performance of the BC and the BBC algorithms without CPLEX cuts

| Set | $\|\Omega\|$ | BC | | | | | | | | BBC | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | with CPLEX cuts | | | | w/o CPLEX cuts | | | | with CPLEX cuts | | | | w/o CPLEX cuts | | | |
| | | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes |
| $\mathcal{S}$ | 100 | 16/16 | 0.0 | 28.8 | 37.8 | 16/16 | 0.0 | 37.0 | 242.1 | 16/16 | 0.0 | 39.9 | 708.3 | 16/16 | 0.0 | 39.9 | 714.9 |
| | 200 | 16/16 | 0.0 | 87.1 | 42.3 | 16/16 | 0.0 | 102.5 | 283.4 | 16/16 | 0.0 | 74.2 | 898.7 | 16/16 | 0.0 | 63.4 | 754.7 |
| | 500 | 16/16 | 0.0 | 487.5 | 50.4 | 16/16 | 0.0 | 522.7 | 307.3 | 16/16 | 0.0 | 145.1 | 881.3 | 16/16 | 0.0 | 155.3 | 828.3 |
| | 1000 | 16/16 | 0.0 | 2031.5 | 58.1 | 16/16 | 0.0 | 2490.1 | 301.9 | 16/16 | 0.0 | 319.5 | 778.3 | 16/16 | 0.0 | 296.5 | 762.8 |
| Average | | 64/64 | 0.0 | 658.7 | 47.1 | 64/64 | 0.0 | 788.1 | 283.7 | 64/64 | 0.0 | 144.7 | 816.6 | 64/64 | 0.0 | 138.8 | 765.2 |
| $\mathcal{L}$ | 100 | 24/24 | 0.0 | 665.4 | 381.4 | 15/24 | 0.8 | 3876.0 | 5035.1 | 10/24 | 1.4 | 5207.6 | 46578.0 | 10/24 | 1.4 | 5033.8 | 44197.3 |
| | 200 | 24/24 | 0.0 | 1775.5 | 354.1 | 12/24 | 1.8 | 4781.5 | 2723.0 | 11/24 | 1.5 | 5375.4 | 38441.7 | 9/24 | 1.5 | 5479.1 | 41797.5 |
| | 500 | 14/24 | $2.0^{(4)}$ | 5063.4 | 168.3 | 4/24 | 4.8 | 6434.8 | 984.9 | 8/24 | 1.8 | 5824.2 | 27722.4 | 8/24 | 1.9 | 5871.3 | 29406.7 |
| | 1000 | 4/24 | $12.9^{(13)}$ | 6889.4 | 24.7 | 0/24 | $5.1^{(11)}$ | 7200.0 | 217.0 | 7/24 | 2.3 | 6246.5 | 20845.3 | 7/24 | 2.3 | 6294.8 | 24777.6 |
| Average | | 66/96 | $3.7^{(17)}$ | 3598.4 | 232.1 | 31/96 | $3.1^{(11)}$ | 5573.1 | 2240.0 | 36/96 | 1.8 | 5663.4 | 33396.9 | 34/96 | 1.8 | 5669.7 | 35044.8 |
| Total | | 130/160 | $2.2^{(17)}$ | 2422.5 | 158.1 | 95/160 | $1.9^{(11)}$ | 3659.1 | 1457.5 | 100/160 | 1.1 | 3455.9 | 20364.8 | 98/160 | 1.1 | 3457.3 | 21332.9 |

$^{(-)}$ the number of instance where a feasible solution could not be found

# 6 Reoptimization Capabilities of the Benders Decomposition Algorithm in Stochastic Environments

The reoptimization capabilities provided by Benders decomposition can be very useful in "what-if" analyses (Geoffrion and Graves 1974, Cordeau et al. 2006). In situations where the changes made to the problem data affect only the master problem, the previously generated Benders cuts are still valid and one can warm-start the algorithm with these cuts. A new optimal solution is typically obtained in a few iterations. Examples of this situation include forcing variables to take specific values or adding logical constraints on the master problem variables. If one instead employs a branch-and-cut algorithm, it must restart from scratch every time a change is made, which can be very time consuming.

Furthermore, if the changes affect only the objective function but not the polyhedron of the Benders dual subproblem, one can also recalculate the Benders cuts from the already generated extreme points. In the two-stage SPRP, changes in demand affect only the objective function of the DFS. Thus, given a new set of demand scenarios, the Benders cuts can be recalculated from the previously identified extreme points and added to the Benders master problem. Instead of retaining Benders cuts, however, one must store the generated extreme points and compute updated Benders cuts corresponding to the new demands. We now discuss how the reoptimization capabilities of the Benders algorithm can be particularly useful in two practical settings, i.e., in a sample average approximation (SAA) scheme and in a rolling horizon (RH) framework.

In both cases, the procedure starts by solving the first problem from scratch using the BBC and keeping the dual solutions generated during the solution process. When the problem is reoptimized, Benders cuts corresponding to the demand scenarios in the new problem are computed using the dual solutions that were already generated in the previous replications. These cuts are added to the Benders master problem before the Benders algorithm starts. To avoid adding too many cuts, which can make the problem size too large, we use a preprocessing step to select the cuts. This process starts by solving the BMP without any Benders cuts to obtain an initial solution $Z^0$ along with the values of $\eta_g$. Then, the cuts with a non-negative left-hand-side value in the initial solution are used as initial Benders cuts. The maximum number of initial Benders cuts is limited to 5000. If the number of cuts exceeds the limit, those with the largest left-hand-side value are selected first. In Table 11, the results obtained by the BBC algorithm with reoptimization are shown in Column *BBC-ReOpt* and the average percentage of the number of initial Benders cuts in each replication is shown in Column *%I.Cuts*. Column *T.CPU* indicates the average total time spent to solve all replications of the same instance and boldface letters are put on the smallest time. Columns *CPU*, *Nodes* and *B.Cuts* are the same as in the previous section and they show the average results per replication.

## 6.1 Reoptimization for a Sample Average Approximation Scheme

Sample average approximation (SAA) (Kleywegt et al. 2002b) is a Monte-Carlo simulation-based sampling method developed to solve problems where the number of scenarios is very large. It can also be applied to problems with continuous distributions or with an infinite number of scenarios. Given a large scenario set, denoted by $\Omega'$, which is intractable, the SAA consists of solving a number of smaller and tractable problems with a sample of size $|\Omega| \ll |\Omega'|$. In the SAA process, one can calculate the SAA gap which is the estimated

Table 11: Performance of the algorithms using the SAA method

| Distribution | $\|\Omega\|$ | $\|M\|$ | $\epsilon$ | BC | | | BBC | | | | BBC-ReOpt | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | T.CPU | CPU | Nodes | T.CPU | CPU | Nodes | B.Cuts | T.CPU | CPU | Nodes | B.Cuts | %I.Cuts |
| Uniform | 100 | 100 | 0.10 | 2703.6 | 27.0 | 35.3 | 3622.7 | 36.2 | 750.0 | 259.6 | **2090.0** | 20.9 | 303.7 | 2571.7 | 97.4 |
| | | | 0.25 | 2777.1 | 27.8 | 35.9 | 3972.9 | 39.7 | 776.7 | 289.1 | **2668.1** | 26.7 | 360.7 | 3135.2 | 97.1 |
| | 200 | 50 | 0.10 | 4348.9 | 87.0 | 44.9 | 3140.4 | 62.8 | 742.4 | 258.6 | **1439.4** | 28.8 | 333.4 | 1904.5 | 96.2 |
| | | | 0.25 | 4373.6 | 87.5 | 43.1 | 3686.3 | 73.7 | 801.1 | 302.0 | **1827.2** | 36.5 | 357.7 | 2427.8 | 96.1 |
| | 500 | 20 | 0.10 | 9596.9 | 479.8 | 52.6 | 2778.8 | 138.9 | 787.8 | 264.4 | **1131.3** | 56.6 | 380.8 | 1007.4 | 91.5 |
| | | | 0.25 | 9372.4 | 468.6 | 50.2 | 3160.5 | 158.0 | 773.9 | 300.1 | **1437.3** | 71.9 | 396.7 | 1287.6 | 91.5 |
| | 1000 | 10 | 0.10 | 18891.8 | 1889.2 | 53.6 | 2531.7 | 253.2 | 819.7 | 268.4 | **1197.4** | 119.7 | 432.5 | 686.3 | 83.6 |
| | | | 0.25 | 19639.4 | 1963.9 | 53.3 | 3223.0 | 322.3 | 808.7 | 339.0 | **1466.7** | 146.7 | 451.2 | 901.4 | 84.7 |
| Average | | | | 8963.0 | 628.9 | 46.1 | 3264.5 | 135.6 | 782.5 | 285.1 | **1657.2** | 63.5 | 377.1 | 1740.3 | 94.5 |
| Normal | 100 | 100 | 0.10 | 2815.3 | 28.2 | 41.8 | 2898.9 | 29.0 | 597.7 | 210.5 | **1664.3** | 16.6 | 235.7 | 2390.4 | 97.6 |
| | | | 0.25 | 2862.2 | 28.6 | 48.4 | 3444.1 | 34.4 | 791.7 | 244.7 | **2331.1** | 23.3 | 368.8 | 2662.7 | 97.1 |
| | 200 | 50 | 0.10 | 4326.1 | 86.5 | 52.7 | 2596.0 | 51.9 | 613.9 | 215.4 | **1044.6** | 20.9 | 243.0 | 1486.1 | 96.4 |
| | | | 0.25 | 4425.6 | 88.5 | 58.1 | 3188.0 | 63.8 | 843.9 | 260.4 | **1639.7** | 32.8 | 416.3 | 1986.4 | 95.9 |
| | 500 | 20 | 0.10 | 9480.4 | 474.0 | 59.7 | 2377.4 | 118.9 | 644.6 | 230.3 | **940.0** | 47.0 | 278.0 | 865.0 | 91.5 |
| | | | 0.25 | 9138.1 | 456.9 | 69.9 | 2712.2 | 135.6 | 839.7 | 260.5 | **1280.2** | 64.0 | 451.6 | 1138.6 | 91.6 |
| | 1000 | 10 | 0.10 | 19107.4 | 1910.7 | 58.2 | 2244.3 | 224.4 | 670.9 | 241.7 | **1008.6** | 100.9 | 317.7 | 634.4 | 84.3 |
| | | | 0.25 | 19069.0 | 1906.9 | 68.6 | 2663.6 | 266.4 | 943.6 | 286.4 | **1257.2** | 125.7 | 450.3 | 760.6 | 84.5 |
| Average | | | | 8903.0 | 622.5 | 57.2 | 2765.6 | 115.5 | 743.2 | 243.7 | **1395.7** | 53.9 | 345.2 | 1490.5 | 94.5 |
| Gamma | 100 | 100 | 0.10 | 2654.1 | 26.5 | 37.1 | 2821.5 | 28.2 | 558.3 | 204.7 | **1536.4** | 15.4 | 213.5 | 2178.1 | 97.6 |
| | | | 0.25 | 2759.2 | 27.6 | 34.6 | 3729.1 | 37.3 | 782.0 | 271.9 | **2385.5** | 23.9 | 338.6 | 3043.5 | 97.3 |
| | 200 | 50 | 0.10 | 4058.4 | 81.2 | 44.3 | 2455.4 | 49.1 | 564.9 | 204.8 | **1022.1** | 20.4 | 213.4 | 1408.3 | 96.4 |
| | | | 0.25 | 4305.7 | 86.1 | 42.4 | 3444.2 | 68.9 | 815.8 | 286.3 | **1705.3** | 34.1 | 373.8 | 2324.9 | 96.2 |
| | 500 | 20 | 0.10 | 8992.3 | 449.6 | 56.5 | 2184.2 | 109.2 | 595.5 | 210.4 | **851.9** | 42.6 | 258.6 | 802.2 | 91.8 |
| | | | 0.25 | 9520.2 | 476.0 | 54.0 | 3027.3 | 151.4 | 826.3 | 292.3 | **1281.0** | 64.1 | 389.4 | 1229.4 | 91.6 |
| | 1000 | 10 | 0.10 | 18456.3 | 1845.6 | 59.0 | 2092.2 | 209.2 | 631.1 | 225.6 | **894.4** | 89.4 | 277.9 | 538.6 | 84.4 |
| | | | 0.25 | 19368.8 | 1936.9 | 57.9 | 2718.1 | 271.8 | 786.4 | 292.9 | **1533.0** | 153.3 | 471.5 | 944.3 | 84.2 |
| Average | | | | 8764.4 | 616.2 | 48.2 | 2809.0 | 115.6 | 695.0 | 248.6 | **1401.2** | 55.4 | 317.1 | 1558.7 | 94.6 |
| Total Average | | | | 8876.8 | 622.5 | 50.5 | 2946.4 | 122.3 | 740.3 | 259.2 | **1484.7** | 57.6 | 346.4 | 1596.5 | 94.5 |

difference between the solution obtained by solving the replications of the sample size $|\Omega|$ and a statistical lower bound on the optimal value for the large scenario set $\Omega'$. This gap can be determined by a sample average function. In practical applications, one can choose a sample size $|\Omega|$ and the number of replications $|M|$ that are most appropriate for the problem in terms of solution quality and computing time. In the SPRP, where different demand scenarios are considered in each replication, one can take advantage of the reoptimization capabilities of Benders decomposition to solve the problem more efficiently. We provide more details on the SAA scheme and steps to estimate the optimality gap in the Appendix.

Tests were performed with sample sizes $|\Omega| = 100$, 200, 500 and 1000 and a number of replications $|M| = 100$, 50, 20 and 10, respectively, which makes the total number of evaluated scenarios equal to 10,000 for every sample size. Because the experiments in this section focus on the benefits of the reoptimization capabilities, we performed the tests on the instance set $\mathcal{S}$ with $l = 3$ and $m = 1$ to avoid excessive computing times. In addition to the uniform distribution, experiments were also performed with normal and gamma distributions with parameters chosen so that the current demand range $[\bar{d}_{it}(1-\epsilon), \bar{d}_{it}(1+\epsilon)]$ corresponds approximately to a 99.5% confidence interval. The size of the large scenario set was chosen equal to $|\Omega'| =$ 10,000 to evaluate the SAA gap. In the first set of experiments, we compared the results provided by different algorithms. Scenarios were generated a priori and all the algorithms were applied to the same scenario sets to ensure a fair comparison. The results are provided in Table 11.

These results clearly indicate the benefits of reoptimization when solving the SPRP in a SAA method. The BBC algorithm using reoptimization could reduce the average total computing time by approximately 50% and 83% compared to the BBC without reoptimization and BC, respectively. The average number of Benders cuts generated by the BBC-ReOpt is significantly larger than for the BBC. However, the majority of them (94.5%) are the initial cuts generated from previous replications and only 5.5% of the number of Benders cuts or 87.7 cuts on average were newly generated at each replication. To further demonstrate the benefits of reoptimization, Figure 2 shows the average computing time spent in each replication to solve the instances with $n = 15$, $\epsilon = 0.10$, $|\Omega| = 200$ and a uniform distribution. The computing times after the first replication are significantly reduced when using the BBC-ReOpt.

To evaluate the impact of the sample size, we further performed experiments to compare the solution quality obtained with different sizes using the BBC-ReOpt. The results are shown in Table 12. These tests follow the procedure presented by Kleywegt et al. (2002b) in which scenarios are generated and the SAA gap is calculated during the SAA process. We report the results at the end of the process after all the replications are solved. Column *ETC (%Abs)* shows the average absolute optimality gap (%) between the expected total costs corresponding to the objective function (1) and the SAA lower bound. Columns *SAA (%Abs)* and ± *SAA (%) at 95% CI* indicate the average absolute optimality gap (%) and the average range of the gap (±%)

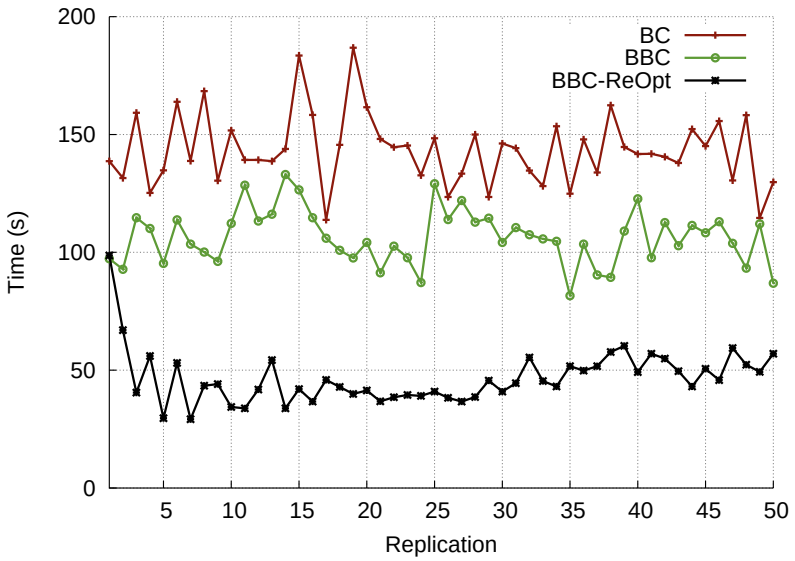


Figure 2: Computing time spent in each replication in the SAA method

Table 12: Solution quality and computing times of the SAA using different sample sizes

| | | ETC (%Abs) | | | | SAA (%Abs) | | | | ± SAA (%) at CI 95% | | | | T.CPU | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Distribution | $\|\Omega\|$ = | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 |
| | $\|M\|$ = | 100 | 50 | 20 | 100 | 100 | 50 | 20 | 100 | 100 | 50 | 20 | 100 | 100 | 50 | 20 | 100 |
| Uniform | $\epsilon$ =0.10 | 0.52 | 0.35 | 0.15 | **0.07** | 0.09 | 0.04 | 0.03 | **0.02** | 0.12 | 0.11 | 0.08 | **0.07** | 1769.1 | 1276.7 | 1113.0 | **1090.6** |
| | 0.25 | 0.98 | 0.59 | 0.29 | **0.13** | 0.13 | 0.12 | 0.04 | **0.04** | 0.25 | 0.23 | 0.17 | **0.14** | 2259.1 | 1616.4 | **1374.2** | 1436.9 |
| | Average | 0.75 | 0.47 | 0.22 | **0.10** | 0.11 | 0.08 | 0.04 | **0.03** | 0.19 | 0.17 | 0.13 | **0.11** | 2014.1 | 1446.6 | **1243.6** | 1263.7 |
| Normal | 0.10 | 0.06 | 0.05 | 0.02 | **0.01** | 0.02 | **0.01** | **0.01** | **0.01** | 0.02 | 0.02 | 0.02 | **0.01** | 1833.2 | 1365.6 | **1224.2** | 1228.4 |
| | 0.25 | 0.10 | 0.06 | 0.04 | **0.03** | 0.02 | 0.02 | **0.01** | **0.01** | 0.03 | 0.02 | 0.02 | **0.01** | 2752.3 | 2079.1 | **1501.0** | 1600.0 |
| | Average | 0.08 | 0.05 | 0.03 | **0.02** | 0.02 | **0.01** | **0.01** | **0.01** | 0.03 | 0.02 | 0.02 | **0.01** | 2292.7 | 1722.3 | **1362.6** | 1414.2 |
| Gamma | 0.10 | 0.03 | 0.02 | **0.01** | **0.01** | 0.01 | 0.01 | **0.00** | **0.00** | 0.02 | **0.01** | **0.01** | **0.01** | 1815.7 | 1315.5 | **1188.5** | 1256.0 |
| | 0.25 | 0.09 | 0.08 | 0.04 | **0.03** | 0.01 | 0.02 | **0.01** | **0.01** | 0.03 | 0.03 | **0.02** | **0.02** | 2662.0 | 2009.1 | **1587.1** | 1801.0 |
| | Average | 0.06 | 0.05 | 0.03 | **0.02** | 0.01 | 0.01 | 0.01 | **0.00** | 0.02 | 0.02 | 0.02 | **0.01** | 2238.8 | 1662.3 | **1387.8** | 1528.5 |
| | Total Average | 0.30 | 0.19 | 0.09 | **0.05** | 0.04 | 0.03 | **0.02** | **0.02** | 0.08 | 0.07 | **0.05** | **0.05** | 2181.9 | 1610.4 | **1331.3** | 1402.2 |

at the 95% confidence interval (CI) using the SAA method, respectively. The gap at the 95% CI is computed with the standard deviation of the optimality gaps obtained by the SAA method and we assume that the gap is normally distributed.

We observe that the largest sample size $|\Omega| = 1000$ can provide the best average ETC gap as well as the best average SAA gap with the least variation compared to the other sample sizes but the average computing time is larger than for the sample size $|\Omega| = 500$. The sample size $|\Omega| = 500$ is, however, generally the best in terms of the trade-off between solution quality and computing time.

## 6.2 Reoptimization for the Dynamic and Stochastic PRP in a Rolling Horizon Framework

In practice, dynamic planning problems are solved repeatedly over time as new information becomes available. This is often done in a rolling horizon framework: a problem is solved using the information for a limited number of periods and in a later period, the problem is solved again using updated information. Usually, there is some overlap between the periods considered in two subsequent problems in order to reduce the end of horizon effect. This version of the SPRP can be seen as a dynamic and stochastic PRP (DSPRP).

In a rolling horizon, one must solve the problem repeatedly with updated initial inventory levels and a new set of demand scenarios. Reoptimization can be particularly useful in this context since the changes do not affect the Benders subproblem polyhedron. To explore the benefits of the reoptimization of the BBC, we performed experiments using the data set used in the previous section but extended to 52 time periods (i.e., one year). We assume that the demand in the current period is more accurate with $\epsilon = 0.1$ and the next periods are more uncertain with $\epsilon = 0.25$. The nominal demand is assumed to increase 5% for every quarter

(i.e., every 13 periods). The number of rolling periods to optimize is set to 3 and 4 and the overlapping period is set to one, which results in a total number of times that the problem has to be solved being equal to $|M|$ = 26 and 17 for 3 and 4 rolling periods, respectively. We assume that the initial inventory for the current rolling periods is equal to the rounded expected value of the corresponding inventory level over all scenarios resulting from the previous rolling periods. The total computing time per instance is limited to 30 hours and the experiments are performed on the instances with $n = 10$ since it is very time consuming to solve larger instance sizes for the whole planning horizon (the total computing time usually exceeds 30 hours).

The results are shows in Table 13 and we indicate the variants by $t_r/t_o$ where $t_r$ and $t_o$ are the number of rolling and overlapping periods, respectively. According to these results, the BBC-ReOpt could reduce the average total computing time by approximately 41% and 81% compared to the BBC without reoptimization and BC, respectively. We can also observe that the reduction in computing time is less for the BBC using reoptimization in the rolling horizon framework compared to the results of the SAA. This is due to the fact that, in each subsequent problem, demands are computed from different nominal cases and initial inventory levels are changed. Thus, the initial cuts generated in the rolling horizon framework are not as strong as in the SAA where demand scenarios of each customer are generated from the same demand distribution.

Table 13: Performance of the algorithms using the rolling horizon method

| $t_r/t_o$ | $\|M\|$ | $\|\Omega\|$ | BC | | | BBC | | | | BBC-ReOpt | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | T.CPU | CPU | Nodes | T.CPU | CPU | Nodes | B.Cuts | T.CPU | CPU | Nodes | B.Cuts | %I.Cuts |
| 3/1 | 26 | 100 | 485.8 | 18.7 | 54.0 | 962.6 | 37.0 | 1361.2 | 365.2 | **439.0** | 16.9 | 567.9 | 1674.4 | 91.3 |
| | | 200 | 1270.4 | 48.9 | 60.4 | 1658.5 | 63.8 | 1385.7 | 377.2 | **863.8** | 33.3 | 585.4 | 1611.6 | 91.0 |
| | | 500 | 6813.9 | 262.1 | 66.9 | 3569.3 | 137.3 | 1418.1 | 380.0 | **1853.7** | 74.0 | 615.7 | 1824.6 | 91.7 |
| | | 1000 | 26577.2 | 1022.2 | 69.5 | 6774.5 | 260.6 | 1300.3 | 395.6 | **3529.7** | 130.6 | 598.8 | 1740.2 | 91.1 |
| Average | | | 8786.8 | 338.0 | 62.7 | 3241.2 | 124.7 | 1366.3 | 379.5 | **1671.6** | 64.3 | 592.0 | 1712.7 | 91.3 |
| 4/1 | 17 | 100 | **889.7** | 52.3 | 133.1 | 2013.3 | 118.4 | 4733.2 | 914.2 | 1376.2 | 81.0 | 2931.0 | 3340.0 | 87.1 |
| | | 200 | 2655.0 | 156.2 | 133.7 | 3177.8 | 186.9 | 3913.4 | 879.3 | **1909.5** | 112.3 | 2474.3 | 3341.1 | 88.9 |
| | | 500 | 15110.2 | 888.8 | 140.3 | 6587.8 | 387.5 | 4024.9 | 862.2 | **4051.2** | 238.3 | 2485.8 | 3555.7 | 88.8 |
| | | 1000 | 65006.9† | 3823.9 | 121.8 | 13537.6 | 796.3 | 4131.4 | 910.7 | **8489.2** | 499.4 | 2478.2 | 3714.9 | 88.1 |
| Average | | | 20915.4† | 1230.3 | 132.2 | 6329.1 | 372.3 | 4200.7 | 891.6 | **3956.5** | 232.7 | 2592.3 | 3487.9 | 88.2 |
| Total | | | 14851.1† | 784.1 | 97.5 | 4785.2 | 248.5 | 2783.5 | 635.6 | **2814.0** | 148.5 | 1592.1 | 2600.3 | 89.8 |

† 15 (out of 68) instances were not solved to optimality and the average optimality gap is 0.4%

# 7 Conclusions

We have addressed demand uncertainty in the production routing problem and proposed Benders reformulations to handle the problem. The Benders algorithms are implemented in a branch-and-cut framework, called branch-and-Benders-cut (BBC), and several computational enhancements, namely, lower bound lifting inequalities, scenario group cuts and Pareto-optimal cuts, are used to improve the performance of the algorithms. The BBC with the best version of the Benders reformulation outperformed the branch-and-cut (BC) algorithm on the standard formulation when solving instances with 500 and 1000 scenarios. The results also indicate that the performance of the BBC does not depend on the CPLEX cuts, while the performance of the branch-and-cut algorithm relies heavily on these cuts. We further discuss reoptimization capabilities in the context of the sample average approximation method and the dynamic and stochastic production routing problem in a rolling horizon framework. The results show that reoptimization can substantially improve the performance of the BBC and BC algorithms for which the computing time is reduced by 50% and 83%, respectively, for the sample average approximation method, and by 41% and 81%, respectively, for the rolling horizon framework.

# Appendices

Section A provides the details of SPRP instances. Section B provides the details of the approach to generate a core point for Pareto-optimal cuts. Section C provides the details of the SAA method.

# A Details of the Instances

The instances of Adulyasak et al. (2012a) for the deterministic PRP were generated from the instances of Archetti et al. (2011) which were designed for heuristics. There are four instances per problem size with

different characteristics as shown in Table 14. We refer to Adulyasak et al. (2012a) for the details of the parameters of the instances. We set the cost of unmet demand $\sigma_i$ proportional to the production and transportation costs, calculated as $\sigma_i = u + 5 \times \text{Round}(f/C + 2c_{0i}/Q)$, where the multiplier 5 is used to ensure that the cost of unmet demand is sufficiently large.

Table 14: Descriptions of the four instances generated from Archetti et al. instances

| No. | Descriptions |
|---|---|
| 1 | Standard instance |
| 2 | High production unit cost, $u\times10$ |
| 3 | Large transportation costs, coordinates$\times5$ |
| 4 | No customer inventory costs |

# B Approach to Determine a Core Point for the Pareto-optimal Cut

We follow a procedure similar to that of Cordeau et al. (2001) to determine a core point for the Pareto-optimal cuts. Recall that $\bar{d}_{it}$ is the nominal demand value and let $\hat{\varepsilon}$ be a positive small value. We solve the problem with one scenario $|\Omega| = 1$ associated with the nominal demand value without the routing variable to determine a core point $\mathbf{y}^0 \in ri(\mathbf{Y}^{LP})$ and $\mathbf{z}^0 \in ri(\mathbf{Z}^{LP})$. The problem is the following.

$$\max \sum_{t\in T} \varrho_t^y + \sum_{i\in N}\sum_{k\in K}\sum_{t\in T} \varrho_{ikt}^z$$

s.t. (2)–(9) and,

$$\begin{aligned}
y_t - \hat{\varepsilon}\varrho_t^y &\geq 0 && \forall t \in T\\
y_t + \hat{\varepsilon}\varrho_t^y &\leq 1 && \forall t \in T\\
z_{ikt} - \hat{\varepsilon}\varrho_{ikt}^z &\geq 0 && \forall i \in N, \forall k \in K, \forall t \in T\\
z_{ikt} + \hat{\varepsilon}\varrho_{ikt}^z &\leq 1 && \forall i \in N, \forall k \in K, \forall t \in T\\
\varrho_t^y, \varrho_{ikt}^z &\in \{0,1\} && \forall i \in N, \forall k \in K, \forall t \in T.
\end{aligned}$$

One can ensure that the core point lies in the relative interior of $\mathbf{Y}^{LP}$ and $\mathbf{Z}^{LP}$. The problem is typically easy to solve and the optimal solution can be obtained in less than one second by using CPLEX. Additionally, it is not necessary to obtain the optimal solution as a feasible solution of the problem is sufficient.

# C Details on Sample Average Approximation (SAA) Method for the SPRP

We adapt the SAA procedure of Kleywegt et al. (2002b) to calculate the approximate optimality gap (SAA gap) as follows.

1. A sample size $|\Omega|$ and the number of replications $|M|$ are determined. A larger sample size $|\Omega'| \gg |\Omega|$ is also selected to compute the estimated objective value of the optimal solution of the SAA problem. The probability of the scenario $\omega$ associated with the sample size $|\Omega|$ is equal to $\rho_\omega = 1/|\Omega|, \forall \omega \in \Omega$
2. For $s = 1 \rightarrow |M|$, do the following steps:
   (a) Generate a sample $\Omega$ and solve the problem to obtain the objective value, denoted by $\nu_\Omega^s$, and the optimal solution, denoted by $Z^s$ (which consists of the vectors $\bar{\mathbf{x}}, \bar{\mathbf{y}}$ and $\bar{\mathbf{z}}$), of the replication $s$.
   (b) Use the solution $Z^s$ to obtain the upper bound of the optimal solution on the generated large sample size $|\Omega'|$, denoted by $\nu_{\Omega'}(Z^s)$, which can be calculated as

$$\nu_{\Omega'}(Z^s) = \sum_{t\in T}\left( f\bar{y}_t + \sum_{(i,j)\in E}\sum_{k\in K} c_{ij}\bar{x}_{ijkt} + \sum_{\omega\in\Omega'}\rho_\omega\left(up_{t\omega} + \sum_{i\in N} h_i I_{it\omega} + \sum_{i\in N_c}\sigma_i e_{it\omega}\right)\right).$$

The variance of this estimated upper bound can be calculated as

$$\sigma^2_{\Omega'}(Z^s) = \frac{1}{|\Omega'|(|\Omega'|-1)} \sum_{\omega \in \Omega'} \left( G_\omega(Z^s) - \nu_{\Omega'}(Z^s) \right)^2$$

where

$$G_\omega(Z^s) = \sum_{t \in T} \left( f\bar{y}_t + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} \bar{x}_{ijkt} + up_{t\omega} + \sum_{i \in N} h_i I_{it\omega} + \sum_{i \in N_c} \sigma_i e_{it\omega} \right).$$

(c) Compute the average of the objective function values obtained in the previous step, denoted by $\hat{\nu}^s_\Omega$, and their corresponding variance as follows:

$$\hat{\nu}^s_\Omega = \frac{1}{s} \sum_{i=1}^{s} \nu^i_\Omega$$

$$\sigma^2_{\hat{\nu}^s_\Omega} = \frac{1}{|M|(|M|-1)} \sum_{i=1}^{s} (\nu^i_\Omega - \hat{\nu}^s_\Omega)^2.$$

The value $\hat{\nu}^s_\Omega$ is a statistical lower bound for the optimal solution value of the sample $\Omega'$.

(d) Compute the SAA gap, denoted by $\varepsilon^{SAA}(\Omega, \Omega')$ and the variance of the gap as follows:

$$\varepsilon^{SAA}(\Omega, \Omega') = \nu_{\Omega'}(Z^s) - \hat{\nu}^s_\Omega$$

$$\sigma^2_{\varepsilon^{SAA}(\Omega,\Omega')} = \sigma^2_{\Omega'}(Z^s) + \sigma^2_{\hat{\nu}^s_\Omega}.$$

The best value of $\varepsilon^{SAA}(\Omega, \Omega')$ and its corresponding $\sigma^2_{\varepsilon^{SAA}(\Omega,\Omega')}$ are kept track of during the process.

3. Choose the solution $Z^s$ that gives the lowest upper bound $\nu_{\Omega'}(Z^s)$ as the best solution.

# References


Adelman, D. 2004. A price-directed approach to stochastic inventory/routing. *Oper. Res.* **52** 499–514.

Adulyasak, Y., J.-F. Cordeau, R. Jans. 2012a. Formulations and branch-and-cut algorithms for multi-vehicle production and inventory routing problems. Tech. Rep. G-2012-14, *Les Cahiers du GERAD*, HEC Montréal, Canada.

Adulyasak, Y., J.-F. Cordeau, R. Jans. 2012b. Optimization-based adaptive large neighborhood search for the production routing problem. *Transportation Sci.* **To Appear**.

Andersson, H., A. Hoff, M. Christiansen, G. Hasle, A. Løkketangen. 2010. Industrial aspects and literature survey: Combined inventory management and routing. *Comput. Oper. Res.* **37** 1515–1536.

Applegate, D., R. Bixby, V. Chvátal, W. Cook. 2011. Concorde TSP solver http://www.tsp.gatech.edu/concorde.html.

Archetti, C., L. Bertazzi, G. Laporte, M. G. Speranza. 2007. A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Sci.* **41** 382–391.

Archetti, C., L. Bertazzi, G. Paletta, M. G. Speranza. 2011. Analysis of the maximum level policy in a production-distribution system. *Comput. Oper. Res.* **38** 1731–1746.

Benders, J. F. 1962. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik* **4** 238–252.

Bertazzi, L., A. Bosco, F. Guerriero, D. Laganà. 2011. A stochastic inventory routing problem with stock-out. *Transport. Res. C-Emer.* **Article in Press**.

Birge, J. R., F. V. Louveaux. 1988. A multicut algorithm for two-stage stochastic linear programs. *Eur. J. Oper. Res.* **34** 384–392.

Birge, J. R., F. V. Louveaux. 2011. Two-stage recourse problems. *Introduction to Stochastic Programming*. Springer Series in Operations Research and Financial Engineering, Springer New York, 181–263.

Codato, G., M. Fischetti. 2006. Combinatorial Benders' cuts for mixed-integer linear programming. *Oper. Res.* **54** 756–766.

Coelho, L. C., J.-F. Cordeau, G. Laporte. 2012. Dynamic and stochastic inventory-routing. CIRRELT Tech. Rep. 2012-37, HEC Montréal, Canada.

Cordeau, J.-F., F. Pasin, M. Solomon. 2006. An integrated model for logistics network design. *Ann. Oper. Res.* **144** 59–82.

Cordeau, J.-F., F. Soumis, J. Desrosiers. 2001. Simultaneous assignment of locomotives and cars to passenger trains. *Oper. Res.* **49** 531–548.

de Camargo, R. S., G. de Miranda Jr., R. P.M. Ferreira. 2011. A hybrid outer-approximation/Benders decomposition algorithm for the single allocation hub location problem under congestion. *Oper. Res. Lett.* **39** 329–337.

de Camargo, R.S., G. de Miranda Jr., H.P. Luna. 2008. Benders decomposition for the uncapacitated multiple allocation hub location problem. *Comput. Oper. Res.* **35** 1047–1064.

Federgruen, A., P. Zipkin. 1984. A combined vehicle routing and inventory allocation problem. *Oper. Res.* **32** 1019–1037.

Fortz, B., M. Poss. 2009. An improved Benders decomposition applied to a multi-layer network design problem. *Oper. Res. Lett.* **37** 359–364.

Gendreau, M., A. Hertz, G. Laporte. 1992. New insertion and postoptimization procedures for the traveling salesman problem. *Oper. Res.* **40** 1086–1094.

Geoffrion, A. M., G. W. Graves. 1974. Multicommodity distribution system design by Benders decomposition. *Management Sci.* **20** 822–844.

Hopp, W., M. Spearman. 2000. *Factory Physics*. 2nd ed. Irwin/McGraw-Hill, NY, USA.

Hvattum, L. M., A. Løkketangen. 2009. Using scenario trees and progressive hedging for stochastic inventory routing problems. *J. Heuristics* **15** 527–557.

Hvattum, L. M., A. Løkketangen, G. Laporte. 2009. Scenario tree-based heuristics for stochastic inventory-routing problems. *INFORMS J. Comput.* **21** 268–285.

Jaillet, P., J. F. Bard, L. Huang, M. Dror. 2002. Delivery cost approximations for inventory routing problems in a rolling horizon framework. *Transportation Sci.* **36** 292–300.

Kleywegt, A. J., V. S. Nori, M. W. P. Savelsbergh. 2002a. The stochastic inventory routing problem with direct deliveries. *Transportation Sci.* **36** 94–118.

Kleywegt, A. J., V. S. Nori, M. W. P. Savelsbergh. 2004. Dynamic programming approximations for a stochastic inventory routing problem. *Transportation Sci.* **38** 42–70.

Kleywegt, A. J., A. Shapiro, T. Homem de Mello. 2002b. The sample average approximation method for stochastic discrete optimization. *SIAM J. Optim.* **12** 479–502.

Lin, S. 1965. Computer solutions of the traveling salesman problem. *Bell Syst. Tech. J.* **44** 2245–2269.

Magnanti, T. L., R. T. Wong. 1981. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Oper. Res.* **29** 464–484.

McDaniel, D., M. Devine. 1977. A modified Benders' partitioning algorithm for mixed integer programming. *Management Sci.* **24** 312–319.

Naoum-Sawaya, J., S. Elhedhli. 2010. An interior-point Benders based branch-and-cut algorithm for mixed integer programs. *Ann. Oper. Res.* **Article in Press** 1–23.

Solyalı, O., J.-F. Cordeau, G. Laporte. 2012. Robust inventory routing under demand uncertainty. *Transportation Sci.* **46** 327–340.

Solyalı, O., H. Süral. 2011. A branch-and-cut algorithm using a strong formulation and an a priori tour-based heuristic for an inventory-routing problem. *Transportation Sci.* **45** 335–345.

Van Slyke, R., R. Wets. 1969. *L*-shaped linear programs with applications to optimal control and stochastic programming. *SIAM J. Appl. Math.* **17** 638–663.