Les Cahiers du GERAD ISSN: 0711–2440

The primal Benders decomposition

E. M. Er Raqabi, I. El Hallaoui, F. Soumis

G–2023–27

August 2023

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2023
– Bibliothèque et Archives Canada, 2023

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2023
– Library and Archives Canada, 2023

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

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

The primal Benders decomposition

El Mehdi Er Raqabi

Issmaïl El Hallaoui

François Soumis

Mathematics and Industrial Engineering Department, Polytechnique Montréal & GERAD, Montréal (Qc), Canada, H3T 1J4

`el-mehdi.er-raqabi@polymtl.ca`

August 2023
Les Cahiers du GERAD
G–2023–27

Abstract : Benders decomposition has been applied significantly to tackle large-scale optimization problems with complicating variables, which, when temporarily fixed, yield problems significantly easier to solve. Still, in its standard form, Benders decomposition, also known as the L-shaped method within the stochastic optimization community, shows several shortcomings. Leveraging the view of Benders decomposition as the dual of Dantzig-Wolfe decomposition, we propose the primal Benders decomposition. This method, a paradigm shift, is based on considering a reduced pool of complicating variables and inserting dynamically promising ones. We show that this method: (i) converges theoretically to optimality, (ii) requires only optimality cuts, referred to as Primal Benders cuts, to reach the optimal solution, (iii) benefits from the integrality of the subproblem, always viewed as a hindrance, and (iv) has an accelerated version with decreasing steps, and for which the number of iterations is, at most, the size of the complicating variables pool. We report promising computational results on the deterministic and stochastic facility location problems for which the proposed method reaches strictly optimal solutions.

Keywords : Benders decomposition, L-shaped method, Dantzig-Wolfe decomposition, mixed-integer programming, large-scale optimization, exact method

Acknowledgements: This work was supported by the Fonds de Recherche du Québec—Nature et Technologies (FRQNT) through a Ph.D. Excellence Scholarship, the Institute for Data Valorisation (IVADO) through another Ph.D. Excellence Scholarship, and the Group for Research in Decision Analysis (GERAD) through computing facilities for the experiments. This support is gratefully acknowledged.

1 Introduction

When tackling large-scale optimization problems, mixed integer linear programming (MILP) has been used intensively as the modeling tool (Guignard-Spielberg and Spielberg, 2005; Lee, 2008). With such usage, intense research has been conducted to tackle MILP problems efficiently (Jünger et al. 2009). Let us consider the MILP problem of the following generic form, referred to as the original problem (OP):

$$\begin{aligned} \mathbf{min}\ \ & f^T y + c^T x \\ s.t.:\ \ & Ay \geq b \\ & Wy + Tx \geq d \\ & y \in \mathbb{Z}_+^n \\ & x \in \mathbb{R}_+^m \end{aligned} \tag{OP}$$

where $f \in \mathbb{R}_+^n$, $c \in \mathbb{R}_+^m$, $A \in \mathbb{R}^{k\times n}$, $b \in \mathbb{R}^k$, $W \in \mathbb{R}^{l\times n}$, $T \in \mathbb{R}^{l\times m}$, and $d \in \mathbb{R}^l$. We can assume without loss of generality that OP is feasible and bounded. Benders decomposition (BD) (Benders, 1962) is a well-known way to tackle problem OP when fixing y implies an easy problem. We refer to y variables as the complicating variables. By projecting OP on the space defined by y variables (Geoffrion, 1970), we obtain:

$$\begin{aligned} \mathbf{min}\ \ & f^T y + \mathbf{min}\{c^T x \mid Tx \geq d - Wy,\ x \in \mathbb{R}_+^m\} \\ s.t.:\ \ & Ay \geq b \\ & y \in \mathbb{Z}_+^n \end{aligned} \tag{OP$_y$}$$

The inner minimization problem is known as the Benders primal subproblem (PSP). Its dual is the Benders dual subproblem (DSP):

$$\begin{aligned} \mathbf{max}\ \ & (d - Wy)^T \lambda \\ s.t.:\ \ & T^T \lambda \leq c \\ & \lambda \geq 0 \end{aligned} \tag{DSP}$$

DSP is preferred to PSP because its polyhedron is independent of the complicating variables y. Solving DSP when fixing $y = \bar{y}$ yields either an extreme point or an extreme ray. Let P and Q be, respectively, the set of extreme points and rays of the DSP polyhedron. The Benders master problem is as follows:

$$\begin{aligned} \mathbf{min}\ \ & f^T y + z \\ s.t.:\ \ & Ay \geq b \\ & z \geq (d - Wy)^T \lambda^p, \quad p \in P \\ & 0 \geq (d - Wy)^T \lambda^q, \quad q \in Q \\ & y \in \mathbb{Z}_+^n \end{aligned} \tag{MP}$$

Enumerating all extreme points and rays is computationally untractable. Thus, the Benders algorithm starts initially with a subset (or empty set) of extreme points and rays. The restricted MP (RMP) is solved, and its solution $\bar{y}$ is provided to DSP. If the latter is feasible and bounded, an optimality cut (corresponding to solution λ^p with $p \in P$) is generated. If it is unbounded, a feasibility cut (corresponding to solution λ^q with $q \in Q$) is generated. These cuts are added to the RMP. Being a relaxation (fewer constraints), the RMP provides a lower bound (LB) on the optimal solution of OP. Also, if feasible, the PSP generates a feasible solution to OP, i.e., an upper bound (UB). The

Benders algorithm continues until the difference between the UB and LB is smaller than a selected threshold $\epsilon \geq 0$.

Given its practical relevance, BD has been applied in many fields, including production routing (Adulyasak et al., 2015), maintenance scheduling (Canto, 2008), healthcare planning (Lin et al., 2016), airline scheduling (Zeighami and Soumis, 2019; Cordeau et al., 2001b), water resource management (Cai et al., 2001), hub location (Contreras et al., 2011), locomotive assignment (Cordeau et al., 2001a), traveling salesman (Laporte et al., 1994), and capacity expansion (Bloom, 1983). Despite its successes, BD is time-consuming, has a zigzagging behavior, and converges slowly. An exhaustive literature review by Rahmaniani et al. (2017) highlights the state of the art of BD application, challenges, and improvement strategies.

Intense research has been conducted to accelerate BD convergence. These efforts can, intuitively, be classified into two sides. The first side deals with improving the LBs provided by the RMP while the second seeks to improve the UBs obtained from the PSP. On the first side, intense research has been developed to select *good* or strengthen Benders cuts (Magnanti and Wong, 1981; Codato and Fischetti, 2006; Fischetti et al., 2010; Sherali and Lunday, 2013; Bodur et al., 2017; Rahmaniani et al., 2020) leading to better LBs. Other acceleration techniques include valid inequalities, warm-starting, managing the branch-and-bound tree, and solving in two phases, *i.e.*, generating first cuts from relaxed MP and then cuts from integer MP. On the second side, as far as we acknowledge, there is no systematic way to generate high-quality UBs. So far, problem-specific heuristics have been used to improve the UBs (Poojari and Beasley, 2009; Rahmaniani et al., 2017).

As highlighted above, accelerating BD convergence requires a double effort to get the lower and upper bounds close to each other as *fast* as possible. In this article, we aim to view BD differently than the commonly established perception (Benders, 1962). Leveraging BD as the dual of Dantzig-Wolfe decomposition (DWD), we propose the Primal Benders Decomposition ($\mathcal{PBD}$). In a nutshell, the $\mathcal{PBD}$ method consists of augmenting the PSP with variables from $supp(\bar{y})$, the support of RMP solution $\bar{y}$ instead of fixing it as in the standard Benders decomposition. In such a case, the BD PSP becomes a restriction of the original problem to a subset of variables. Using the solutions of the BD PSP, we generate Benders cuts for the BD RMP. We then solve the latter to identify *promising* complicating variable(s) y to be added to the BD PSP. We augment the BD PSP iteratively with these variables until convergence. This is very relevant in large-scale contexts where companies keep a history of *good* solutions. These solutions can be provided as an initial point and thus only a few complicating variables might be required and inserted for convergence.

The main contributions of this paper are the following: (1) Proposing the $\mathcal{PBD}$ method and proving its convergence, (2) Highlighting that its accelerated version reaches optimal or near-optimal solutions in at most the size of the complicating variables (n iterations at most) with decreasing steps (no zigzagging), (3) Generating only *interesting* optimality cuts, referred to as Pareto-optimal primal Benders cuts, (4) Showing that handling the integrality at the subproblem level is no longer an issue when using $\mathcal{PBD}$ and that solving to optimality the integer RMP is no longer required, and (5) Testing the proposed method on various instances of the facility location problem, for which it reaches linearly and strictly optimal solutions.

The remainder of this article is organized as follows. In Section 2, we present the $\mathcal{PBD}$ method, and in Section 3, we design an accelerated version of it. Section 4 provides a facility location example to illustrate the method. The experimental design and computational results are provided in Sections 5 and 6, respectively. We conclude in Section 7.

2 The Primal Benders Decomposition

In this section, we present the motivation behind the $\mathcal{PBD}$ method development. Then, we highlight the $\mathcal{PBD}$ framework and its convergence.

2.1 Motivation

In the literature, the main trend is the dual view of BD (visualized in Figure 1), where the BD RMP provides part of the solution (primal information) to the BD SP (i.e., BD PSP). The latter completes the primal information and uses its dual solution to generate Benders cuts (row(s)) for the BD RMP. Such a view has driven research in BD since early developments of the method (Benders, 1962; Rahmaniani et al., 2017).

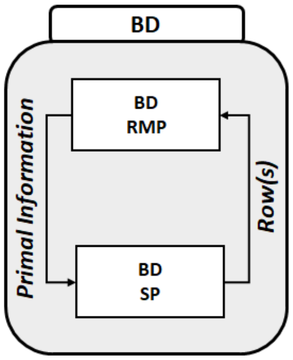


Figure 1: Standard view of Benders Decomposition

Another interesting view consists of perceiving the BD as the dual of the DWD. From such a perspective, the BD RMP corresponds to the DWD SP, while the BD SP corresponds to the DWD RMP. In the DWD, the DWD RMP is used to provide the DWD SP with the necessary dual information (dual solutions) from which the DWD SP generates improving (if they exist) columns for the DWD RMP. While the latter accumulates the columns (the UB decreases because the domain grows), the BD SP uses only the last column generated, leading to the UB zigzagging behavior. Also, while the BD RMP accumulates the dual information (the LB increases because the domain is reduced), the DWD SP uses only the dual information of the last DWD RMP solution, leading to the LB zigzagging behavior (Valério de Carvalho, 2005). The LB of BD RMP is dual and resembles, in this sense, the lower bound obtained with the reduced cost in the SP of DWD. Conversely, the BD SP and DWD RMP give an UB (primal). In practice, we prefer more primal solutions since they inform the decisions to be implemented. Thus, the problem (BD SP in our context) that provides primal solutions should receive more attention, in our opinion. We highlight this view in Figure 2.

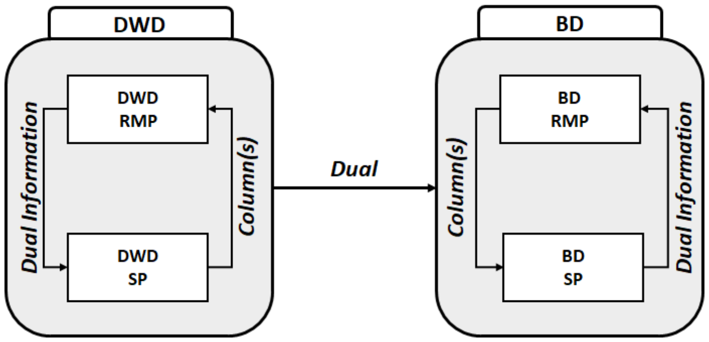


Figure 2: BD as the dual of DWD

The presented insight allows the design of the $\mathcal{PBD}$. In the latter, the RMP inserts improving column(s) into the SP. Then, the SP generates Benders cut(s) for the RMP. In such a way, the $\mathcal{PBD}$ accumulates information in both the RMP and SP and improves monotonically the lower and upper bounds. Based on the motivation, we present next the $\mathcal{PBD}$ framework.

2.2 The PBD framework

Given a pool of complicating variables y, we generate an initial solution y^{init}. In practice, we may already have a good initial solution either from the company's history of solutions using machine learning or in case of re-optimization after perturbation (Er Raqabi et al., 2023b). The support of the initial solution, referred to as $supp(y^{init})$, is added to a set $\mathcal{S}$ (initially empty), and the corresponding complicating variables (y_j with $j \in \mathcal{S}$) are inserted into the reduced primal subproblem (RPSP$_\mathcal{S}$). Since we do not consider all the y variables, we use the qualification *reduced*.

Formally, let $y_\mathcal{S} \in \mathbb{Z}_+^n$ be the vector of complicating variables and let $y_j, j \in J = \{1, 2, ..., n\}$ be an element of it such that $y_j = 0$ if $j \notin \mathcal{S}$. We formulate the RPSP$_\mathcal{S}$ as follows:

$$\begin{aligned} \mathbf{min}\ \ & f^T y_\mathcal{S} + c^T x \\ s.t.:\ \ & A y_\mathcal{S} \geq b \\ & W y_\mathcal{S} + T x \geq d \ \ [\lambda] \\ & y_\mathcal{S} \in \mathbb{Z}_+^n \\ & x \geq 0 \end{aligned} \tag{RPSP$_\mathcal{S}$}$$

By fixing $y_\mathcal{S}$ to $\bar{y}_\mathcal{S}$, a feasible solution of RPSP$_\mathcal{S}$ and by denoting λ the vector of dual variables corresponding to the second set of constraints, we obtain the Reduced Dual Subproblem (RDSP$_\mathcal{S}$):

$$\begin{aligned} \mathbf{max}\ \ & (d - W\bar{y}_\mathcal{S})^T \lambda \\ s.t.:\ \ & T^T \lambda \leq c \\ & \lambda \geq 0 \end{aligned} \tag{RDSP$_\mathcal{S}$}$$

The $\mathcal{PBD}$ framework is highlighted in Figure 3. Using y^{init} as a warm-start, we first solve RPSP$_\mathcal{S}$ to optimality. From the explored leaf nodes of the Branch & Bound (B&B), we generate a pool of Benders cuts using each node's dual solution. We add these cuts to the BD RMP, which is solved to optimality. Let $\bar{y}$ be its solution. The BD RMP provides a LB on the optimal value of OP. Also, the RPSP$_\mathcal{S}$ provides a feasible solution to OP, i.e., an UB on the optimal value of OP. If $|UB - LB| \geq \epsilon$, the new complicating variables in $supp(\bar{y})$ (i.e., $supp(\bar{y}) \nsubseteq \mathcal{S}$ with $\bar{y}$ being the BD RMP's solution) are inserted into the RPSP$_\mathcal{S}$ ($S = S \cup supp(\bar{y})$). The algorithm continues until the difference between the UB and LB is smaller than a selected threshold $\epsilon \geq 0$. In such a case, we return the optimal solution $(y^*_\mathcal{S}, x^*)$ of RPSP$_\mathcal{S}$, which is the optimal solution of OP after augmenting it with zeros. From one iteration to another, the UB decreases because the domain grows and the LB increases because the domain is reduced. The solving of the RPSP$_\mathcal{S}$ should be *fast* if we warm-start using its previous solution and $\bar{y}$.

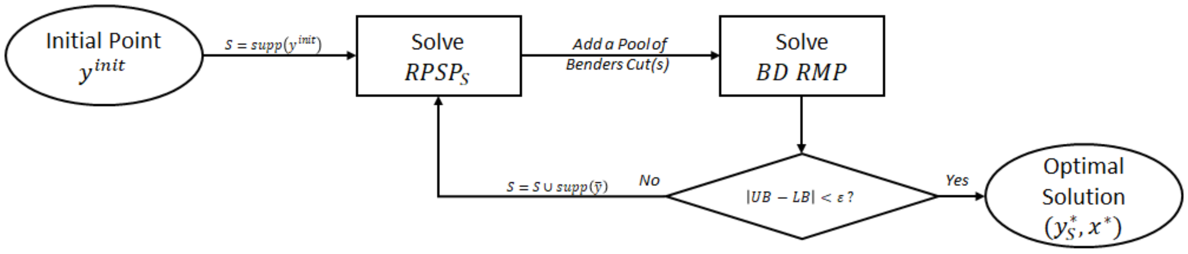


Figure 3: The PBD framework

The complicating variables added to the RPSP$_\mathcal{S}$ are not fixed, as in BD SP. Thus, the RPSP$_\mathcal{S}$ is a restriction of OP and this is why $\mathcal{PBD}$ is primal because we improve the current integer solution at each iteration. We discuss next the convergence of the $\mathcal{PBD}$ method.

2.3 The PBD convergence

In this section, we discuss the $\mathcal{PBD}$ method convergence. We start with an observation related to the cuts obtained by the standard BD.

Observation 1. Let $y \in \mathbb{N}^n$ be a vector of complicating integer variables. From this vector, we can construct $\aleph_0^n$ solutions $\bar{y}$. Thus, in the worst case, the standard BD may generate $\aleph_0^n$ cuts before convergence, i.e., explore all extreme points and rays of DSP.

Observation 1 is the main insight behind the slow convergence and the zigzagging behavior of BD, especially in large-scale contexts. This is similar to the DWD case where *oscillations* are observed (Ben Amor et al., 2006). In the BD case, it is possible to move from a good BD RMP solution to a much worse one. This affects the quality of the UB obtained by the $\text{RPSP}_{\mathcal{S}}$ in the following iteration. Thus, we may wonder whether all the BD RMP solutions are relevant to reach the optimal solution. If this is not the case, then we may seek to identify just the relevant solutions. One way to confirm is finding the *best* $\bar{y}$ as sketched in the example below.

Example 1. Given a vector $y \in \mathbb{B}^2$, instead of providing $\bar{y} = (0,0)$, $\bar{y} = (1,0)$, $\bar{y} = (0,1)$, and $\bar{y} = (1,1)$ to the Benders subproblem as in BD SP, we provide the variables $y = (y_1, y_2)$ (i.e., $\mathcal{S} = \{1,2\}$). Then, solving the $\text{RPSP}_{\mathcal{S}}$ via B&B will provide the optimal (the best) $\bar{y}$.

The Observation 1 and the Example 1 above align with the $\mathcal{PBD}$ motivation and design. Viewing the $\text{RPSP}_{\mathcal{S}}$ from a B&B perspective leads us to generate the following result.

Proposition 1. On the B&B tree of $\text{RPSP}_{\mathcal{S}}$, each integer feasible node allows generating a Benders' optimality cut, while each infeasible node allows generating a Benders' feasibility cut. Furthermore, this cut can be obtained using the node's dual solution.

Proof. Within the B&B tree, we distinguish feasible and infeasible nodes. Within the feasible nodes, we consider integer nodes. Assuming that no cuts are added and that the branching is standard on each variable alone (not on a subset of variables), the $\text{RPSP}_{\mathcal{S}}^{Node}$ at an integer node is written as:

$$\begin{aligned} \textbf{min} \quad & f^T y_{\mathcal{S}} + c^T x && (\text{RPSP}_{\mathcal{S}}^{Node}) \\ s.t.: \quad & A y_{\mathcal{S}} \geq b \\ & W y_{\mathcal{S}} + T x \geq d \quad [\lambda] \\ & y_{\mathcal{S}} = \bar{y}_{\mathcal{S}} \\ & x \geq 0 \end{aligned}$$

Constraints $y_{\mathcal{S}} = \bar{y}_{\mathcal{S}}$ follow from the branching constraints. By fixing $y_{\mathcal{S}} = \bar{y}_{\mathcal{S}}$ in $\text{RPSP}_{\mathcal{S}}^{Node}$ and removing constraints $y_{\mathcal{S}} = \bar{y}_{\mathcal{S}}$, we obtain BD PSP. Thus, we generate the same Benders optimality cut using the integer node's dual solution. In a similar way, we prove that we generate a Benders feasibility cut using an infeasible node's dual solution. □

We refer to the Benders cuts obtained from the B&B leaf nodes' dual solutions as the primal Benders cuts. From the B&B perspective, an interesting observation follows.

Observation 2. In the B&B process, some of the nodes will be pruned, thus eliminating several irrelevant solutions $\bar{y}_{\mathcal{S}}$ in the corresponding leaves, and consequently their corresponding Benders cuts.

The pruned nodes are *dominated* by the unpruned ones. It implies that the *pruned* Benders cuts (corresponding to pruned nodes) are *dominated* by the *unpruned* Benders cuts (corresponding to unpruned nodes). Thus, pruning allows reducing the number of Benders cuts obtained throughout the $\mathcal{PBD}$ framework. It is not the case for BD, which may generate all Benders cuts, thus significantly increasing the number of iterations and the size of the BD RMP. In the next lemma, we show that the primal Benders cuts are valid for the BD RMP.

Lemma 1. The primal Benders cut corresponding to an integer feasible or an infeasible node of RPSP$_{\mathcal{S}}$ is valid for the BD RMP.

The proof of Lemma 1 is straightforward (see Proposition 1). Lemma 1 implies that no cut lifting is needed. Another strength of the $\mathcal{PBD}$ is highlighted in the observation below.

Observation 3. Once the RPSP$_{\mathcal{S}}$ feasibility and boundness are ensured from the initial iteration, it remains feasible and bounded (we enrich the subproblem with new variables) throughout iterations.

The next proposition shows that when there are no more complicating variables to insert in the RPSP$_{\mathcal{S}}$, the UB and LB coincide.

Proposition 2. Let $(\bar{y},\bar{z})$ be the solution of RMP at iteration $t \in \mathbb{N}^*$. If $supp(\bar{y}) \subseteq \mathcal{S}$, then $UB = LB$.

Proof. Suppose that $UB > LB$. Since $supp(\bar{y}) \subseteq \mathcal{S}$, we distinguish two cases: optimality or feasibility cut. Let us discuss the optimality cut case (the same reasoning applies to the feasibility cut case). If $\bar{y}$ is feasible for RPSP$_{\mathcal{S}}$, the optimality cut corresponding to $\bar{y}$ (valid for the RMP as per Lemma 1) at iteration $t-1$ with $\bar{\lambda}$ the dual solution of DSP is:

$$z \geq (d - Wy)^T \bar{\lambda}$$

with $z = \hat{z}$ being the RMP solution at iteration $t-1$, we have:

$$\hat{z} \geq (d - W\bar{y})^T \bar{\lambda}$$

Since $UB > LB$, the optimality cut above is such that at iteration t:

$$\bar{z} = (d - W\bar{y})^T \bar{\lambda} > \hat{z}$$

Contradiction. □

An interesting result follows from Proposition 2.

Corollary 1. The $\mathcal{PBD}$ does not generate the same pool of Benders cut(s) twice.

The proof is straightforward since generating the same pool of Benders cut(s) implies obtaining the same $\bar{y}$ for the RMP, i.e., $supp(\bar{y}) \subseteq \mathcal{S}$. Next, we prove the $\mathcal{PBD}$ convergence.

Theorem 1. The $\mathcal{PBD}$ method converges.

Proof. As per Lemma 1, the Benders cuts computed in the B&B tree of RPSP$_{\mathcal{S}}$ are valid for the BD RMP. As per Corollary 1, the $\mathcal{PBD}$ does not generate the same pool of Benders cut(s) twice. Given that the number of Benders cuts is finite (because the number of extreme points and rays is finite), the $\mathcal{PBD}$ converges. □

Under its basic form, the $\mathcal{PBD}$ method may insert unpromising complicating variables. This implies solving potentially large MILP subproblems and master problems at each iteration, and consequently a large execution time. The latter is increased further in the context of large-scale optimization. The larger the master problem and the subproblem(s), the larger the execution time. Thus, to make it efficient, we design an accelerated $\mathcal{PBD}$ version, which is presented next.

3 The accelerated PBD

In this section, we present the acceleration strategies, the accelerated $\mathcal{PBD}$ algorithm, and its convergence. We highlight in this section that optimality cuts are sufficient to reach optimal or near-optimal (primal) solution(s).

3.1 Acceleration strategies

In this section, we discuss some tips allowing the efficient implementation of the $\mathcal{PBD}$ method. These acceleration tips are classified into two aspects: master problem and subproblem acceleration strategies.

3.1.1 Master problem acceleration strategies

Solving both the integer master problem and the integer subproblem is very costly. To tackle such a burden, we shift integrality to the subproblem and tackle the master problem in its relaxed (integrality) form. For the master problem, we distinguish two acceleration strategies: selection strategy and subsets.

Selection Strategy. To alleviate the BD RMP, it is not necessary to consider all y variables. We may, similarly to the RPSP$_{\mathcal{S}}$, insert the promising complicating variables gradually. In such a case, the reduced restricted master problem (RRMP$_{\mathcal{T}}$) can be written as:

$$\begin{aligned} \textbf{min} \quad & f^T y_{\mathcal{T}} + z & & \text{(RRMP}_{\mathcal{T}}\text{)} \\ s.t.: \quad & A y_{\mathcal{T}} \geq b & [\alpha] & \\ & z \geq (d - W y_{\mathcal{T}})^T \lambda^p, \;\; p \in P & [\beta] & \\ & y_{\mathcal{T}} \in \mathbb{Z}_+^n & & \end{aligned}$$

Similarly to RPSP$_{\mathcal{S}}$, $y_{\mathcal{T}} \in \mathbb{Z}_+^n$ is the vector of $y_j, j \in J = \{1, 2, ..., n\}$ such that $y_j = 0$ if $j \notin \mathcal{T}$. The motivation behind the consideration of extreme points without extreme rays follows from the feasibility of RPSP$_{\mathcal{S}}$ (Observation 3). When the RRMP$_{\mathcal{T}}$ is relaxed (integrality), let α and β_p, $p \in P$ be the dual solutions vectors corresponding to its constraints. Also, let a_j be column j of A and w_j be column j of W where $j \in \{1, 2, ..., n\}$. The reduced cost formula corresponding to variable y_j with $j \in J \setminus \mathcal{T}$ is:

$$\bar{f}_j = f_j - a_j^T \alpha - \sum_{p \in P} COEF(j,p)\beta_p, \;\; j \in J \setminus \mathcal{T} \tag{1}$$

The goal is to find promising complicating variables, i.e., y_j, $j \in J \setminus \mathcal{T}$ such that $\bar{f}_j < 0$, and select a few to be added to the RRMP$_{\mathcal{T}}$. For each variable y_j, $j \in J \setminus \mathcal{T}$ (not present in the RRMP$_{\mathcal{T}}$), the computation of reduced cost $\bar{f}_j$ requires the computation of this variable's coefficients $COEF(j,p)$, $p \in P$ in the already existing primal Benders cuts in RRMP$_{\mathcal{T}}$.

Proposition 3. Let $p \in P$. The coefficient of a potential variable $\phi \in J \setminus \mathcal{T}$ in the Benders cut corresponding to extreme point p is $COEF(\phi,p) = w_\phi^T \lambda^p$.

Proof. Let $\mathcal{S}^+ = \mathcal{S} \cup \{\phi\}$, $\phi \in J \setminus \mathcal{T}$. In the DSP, we have $\bar{y}_{\mathcal{S}^+} = \bar{y}_{\mathcal{S}}$ because $\bar{y}_\phi = 0$. Thus, the DSP's solution remains the same, i.e., λ^p. From $(d - W y_{\mathcal{S}^+})^T \lambda^p$, we infer that $COEF(\phi,p) = w_\phi^T \lambda^p$. □

As per Proposition 3, the computation of reduced costs is quick since λ^p, $p \in P$ are already computed.

Subsets. The subset of complicating variables in RRMP$_{\mathcal{T}}$ contribute to only a subset of constraints implying a subset of x variables. Some constraints become redundant. We can remove them by preprocessing. It is worth mentioning that, to differentiate between the LB obtained by the RMP and the LB obtained by the RRMP$_{\mathcal{T}}$, we refer to the latter as reduced LB (RLB). Since the RRMP$_{\mathcal{T}}$ contains fewer variables, its RLB is not necessarily a LB for OP.

This completes the acceleration strategies for the RMP. We can now present the acceleration strategies for the RPSP$_{\mathcal{S}}$.

3.1.2 Subproblem acceleration strategies

For the subproblem, the acceleration strategies are classified into local Pareto-optimal cuts, warm-start, and subsets.

Local Pareto-optimal Cuts. When solving RPSP$_\mathcal{S}$, all the integer solutions $\bar{y}$ explored in the B&B tree can be collected and corresponding Benders optimality cuts can be obtained. Adding several cuts to the RRMP$_\mathcal{T}$ may increase execution time. Instead of inserting all the Benders cuts corresponding to the identified $\bar{y}_\mathcal{S}$ solutions, we seek the relevant ones. Inspired by the notion of Pareto-optimal cuts (Magnanti and Wong, 1981), we introduce the notion of *local* Pareto-optimal cuts.

Definition 1. A cut

$$z \geq (d - Wy_\mathcal{S})^T \bar{\lambda}$$

is dominated locally by

$$z \geq (d - Wy_\mathcal{S})^T \lambda^*$$

if

$$(d - Wy_\mathcal{S})^T \bar{\lambda} \leq (d - Wy_\mathcal{S})^T \lambda^* \qquad \forall y_\mathcal{S} \in \mathbb{Z}_+^n$$

and there exist a $\bar{y}_\mathcal{S} \in \mathbb{Z}_+^n$ such that

$$(d - W\bar{y}_\mathcal{S})^T \bar{\lambda} < (d - W\bar{y}_\mathcal{S})^T \lambda^*$$

A cut is locally Pareto-optimal if it is not locally dominated by any other cut.

Following the definition, we show that the Benders optimality cut(s) obtained from the optimal node(s) in the B&B of RPSP$_\mathcal{S}$ are locally Pareto-optimal.

Proposition 4. In the B&B Tree of RPSP$_\mathcal{S}$, the Benders cut obtained from the optimal node is locally Pareto-optimal.

Proof. Let $(x^*_\mathcal{S}, y^*_\mathcal{S})$ be the optimal solution of RPSP$_\mathcal{S}$. When fixing $y_\mathcal{S} = y^*_\mathcal{S}$ in RPSP$_\mathcal{S}$ and moving to the dual, let λ^* be the dual optimal solution corresponding to $(x^*_\mathcal{S}, y^*_\mathcal{S})$. Then, the Benders cut corresponding to the optimal node is the following:

$$z \geq (d - Wy_\mathcal{S})^T \lambda^* \tag{2}$$

Suppose that this cut is dominated by another cut corresponding to a non-optimal node. Then by Definition 1, there exists $\bar{\lambda}$ such that:

$$(d - Wy_\mathcal{S})^T \lambda^* \leq (d - Wy_\mathcal{S})^T \bar{\lambda} \qquad \forall y_\mathcal{S} \in \mathbb{Z}_+^n$$

For $y_\mathcal{S} = y^*_\mathcal{S}$, we have:

$$(d - Wy^*_\mathcal{S})^T \lambda^* \leq (d - Wy^*_\mathcal{S})^T \bar{\lambda}$$

Given that λ^* is a dual optimal solution of RDSP$_\mathcal{S}$, we also have:

$$(d - Wy^*_\mathcal{S})^T \bar{\lambda} < (d - Wy^*_\mathcal{S})^T \lambda^*$$

Contradiction. □

Based on Propositions 4, we insert only the primal Benders optimality cut(s) corresponding to the optimal node(s), which are locally Pareto-optimal. These cuts capture all the necessary information to be provided to the RRMP$_{\mathcal{T}}$ at each iteration. In such a way, the master problem has fewer constraints, and the solving is faster. The next result follows.

Corollary 2. If $y^*_{\mathcal{S}}$ augmented by zeros is optimal for the *OP*, the corresponding Benders cut is Pareto-optimal.

The proof is straightforward since the local cut is valid for the OP and is Pareto-optimal using the same proof as Proposition 4.

Warm-start. RPSP$_{\mathcal{S}}$ is a MILP, implying that solving it from scratch might be costly. Thus, warm-starting is an effective way to tackle it efficiently. To accelerate the RPSP$_{\mathcal{S}}$ solving, we may warm-start it at each iteration. Let RPSP$_{\mathcal{S}+}$ be RPSP$_{\mathcal{S}}$ augmented with a new complicating variable provided by the selection strategy, we have the following observation.

Observation 4. Let $(x^*_{\mathcal{S}}, y^*_{\mathcal{S}})$ be the optimal solution of RPSP$_{\mathcal{S}}$. When augmenting RPSP$_{\mathcal{S}}$ with an improving complicating variable y_ϕ, a basic feasible solution for RPSP$_{\mathcal{S}+}$ is obtained when $y^*_{\mathcal{S}}$ is augmented with a zero corresponding to variable y_ϕ and $x^*_{\mathcal{S}}$ is augmented with zero(s) corresponding to the new x variable(s) added to RPSP$_{\mathcal{S}}$. Furthermore, the integrality gap is much smaller than if we consider all the variables y and x of the OP.

Warm-starting allows the $\mathcal{PBD}$ to benefit from the primal information to close the gap and reach optimality quickly. Furthermore, we recall that, at each iteration, we insert only the most promising variables identified using Formula 1. This allows keeping the RPSP$_{\mathcal{S}}$ as small as possible.

Subsets. Since not all the complicating variables y are present in the RPSP$_{\mathcal{S}}$, some constraints become redundant as well as some x variables, we may reduce the size of the model by removing these variables and constraints. For instance, let us consider the constraint of the form $\sum_{i=1}^{m} x_{ij} \leq y_j \ \ \forall j \in J$, with all variables being positive. For a given $k \in J$, if $y_k = 0$ then $x_{ik} = 0 \ \ \forall i \in \{1, 2, ..., m\}$ and constraint $\sum_{i=1}^{m} x_{ik} \leq y_k$ is redundant.

This completes the acceleration strategies for the RPSP$_{\mathcal{S}}$. We present next the Accelerated $\mathcal{PBD}$ algorithm.

3.2 The Accelerated PBD algorithm

The Accelerated $\mathcal{PBD}$ algorithm is summarized in Algorithm 1. Let y^{init} be an initial point. We set $supp(y^{init})$ to sets $\mathcal{S}$ and $\mathcal{T}$, and the current solution y^{init} to $\bar{y}$. Then, we solve RPSP$_{\mathcal{S}}$ to optimality. We generate the local Pareto-optimal primal Benders cut(s) using the optimal node(s)' dual solution(s). We add these cuts to the RRMP$_{\mathcal{T}}$, which is relaxed (integrality) and solved. Let $\bar{y}$ be its new solution. If the solution changes ($supp(\bar{y})$ changes), we add $supp(y^*)$ to $\mathcal{S}$ and insert the new complicating variables into RPSP$_{\mathcal{S}}$. We solve the latter, and the process continues. Otherwise (the solution does not change), using RRMP$_{\mathcal{T}}$'s dual solution, we compute the reduced cost for each y_j with $j \in J \setminus \mathcal{T}$. If we identify promising complicating variables, we select a few, add their indexes to set $\mathcal{T}$, and insert them into the RRMP$_{\mathcal{T}}$ by lifting the existing Benders cuts. We solve the new RRMP$_{\mathcal{T}}$. The process continues until no improving complicating variable is identified ($\Phi = \emptyset$). In such a case, we run Algorithm 2 to check convergence (detailed in Section 3.3). Algorithm 2 returns the optimal solution $(y^*_{\mathcal{S}}, x^*_{\mathcal{S}})$ of RPSP$_{\mathcal{S}}$, which is augmented with zeros to obtain an optimal or near-optimal solution of OP.

3.3 The Accelerated PBD convergence

A feasible solution of OP can be constructed by taking the RPSP$_{\mathcal{S}}$ solution and augmenting it with zeros corresponding to the remaining variables (present in OP and not in RPSP$_{\mathcal{S}}$). The following

Algorithm 1: Accelerated PBD

```
1  Generate an initial point $y^{init}$
2  $\mathcal{S} \leftarrow supp(y^{init})$, $\mathcal{T} \leftarrow supp(y^{init})$, $\Phi \leftarrow \emptyset$, $\bar{y} \leftarrow y^{init}$
3  Solve RPSP$_\mathcal{S}$ to optimality, generate primal Benders optimality cut(s) at optimal B&B node(s), and insert
     these cuts into RRMP$_\mathcal{T}$
4  Solve relaxed RRMP$_\mathcal{T}$ to get new solution $\bar{y}$
5  if $supp(\bar{y})$ changes then
6      $\mathcal{S} \leftarrow \mathcal{S} \cup supp(\bar{y})$
7      Return to Line 3
8  else
9      Using relaxed RRMP$_\mathcal{T}$'s dual solutions, compute $\bar{f}_j$ for each $y_j$ with $j \in J \setminus \mathcal{T}$ as per Formula 1
10     $\Phi \leftarrow \{j \in J \setminus \mathcal{T} : \bar{f}_j < 0\}$
11     if $\Phi \neq \emptyset$ then
12         Select few variables $y_j$, $j \in \Phi$ and add their indexes to set $\mathcal{T}$
13         Lift Benders cuts in RRMP$_\mathcal{T}$ using the coefficients $COEF(j,p)$, $p \in P$
14         Return to Line 4
15     end
16 end
17 Run Algorithm 2 to check convergence
```

theorem highlights that the Accelerated $\mathcal{PBD}$ provides a finite decreasing sequence to an optimal or near-optimal solution of OP.

Theorem 2. If we start with an initial point (x^1, y^1) and execute Algorithm 1, we generate a sequence $(x^1, y^1), (x^2, y^2), ..., (x^h, y^h)$ of solutions such that:

1. $f^T y^1 + c^T x^1 \geq f^T y^2 + c^T x^2 \geq ... \geq f^T y^h + c^T x^h$;
2. $(x^1, y^1), (x^2, y^2), ..., (x^h, y^h)$ are solutions of OP;
3. (x^h, y^h) is an optimal or near-optimal solution of OP.

Proof. From any feasible solution of RPSP$_\mathcal{S}$, we can obtain a feasible solution for OP. Furthermore, when augmenting RPSP$_\mathcal{S}$ with a variable y_ϕ with $\phi \in J \setminus \mathcal{S}$, we have $f_\mathcal{S}^T y_\mathcal{S}^* + c^T x_\mathcal{S}^* \geq f_{\mathcal{S}^+}^T y_{\mathcal{S}^+}^* + c_{\mathcal{S}^+}^T x_{\mathcal{S}^+}^*$. Combining the above, we form a decreasing sequence $f^T y^1 + c^T x^1 \geq f^T y^2 + c^T x^2 \geq ... \geq f^T y^h + c^T x^h$ where $(x^1, y^1), (x^2, y^2), ..., (x^h, y^h)$ are solutions of OP and where the last solution (x^h, y^h) is an optimal or near-optimal solution of OP because either there are no more complicating variables with a negative reduced cost to insert or all the complicating variables are inserted, i.e., RPSP$_\mathcal{S}$ is equivalent to OP. □

The following observation highlights the significant reduction in the number of Benders iterations between the standard BD and the Accelerated $\mathcal{PBD}$.

Observation 5. Compared to BD for which $\aleph_0^n$ iterations may be required in the worst case, the Accelerated $\mathcal{PBD}$ requires at most n iterations in the worst case. In such a case, the RPSP$_\mathcal{S}$ becomes OP because n is the number of complicating variables in the OP and is also the number of variables in J.

In the standard BD, each cut is obtained from a solution $\bar{y}$. For $y \in \mathbb{Z}_+^n$, we obtain $\aleph_0^n$ solutions to explore in the worst case, i.e., $\aleph_0^n$ iterations. In the $\mathcal{PBD}$ case, each complicating variable with a negative reduced cost induces a single iteration. Thus, n iterations are required in the worst case, i.e., a reduction from $\aleph_0^n$ to n. It follows that, if $y \in \{0,1\}^n$, in the worst case, the number of Benders iterations is reduced from 2^n to n when using the Accelerated $\mathcal{PBD}$ instead of BD.

To confirm convergence to optimality, we proceed as follows. Once no complicating variable(s) with a negative reduced cost are identified or there is no change in the optimal solution of RRMP$_\mathcal{T}$, we insert all remaining complicating variables into RRMP$_\mathcal{T}$ using the previously computed lifting coefficients. Then, we solve the RRMP$_\mathcal{T}$ and compute its objective value, which becomes a LB for the OP problem (RRMP$_\mathcal{T}$ contains, at this stage, all complicating variables and becomes RMP). If

$|UB - LB| < \epsilon$, we return the optimal solution $(y^*_{\mathcal{S}}, x^*)$ of RPSP$_{\mathcal{S}}$, which is augmented with zeros to obtain the optimal solution of OP. Otherwise, we continue the insertion of complicating variables into RRMP$_{\mathcal{T}}$ as per the framework in Figure 3 until satisfying $|UB - LB| < \epsilon$. In such a case, we return the optimal solution $(y^*_{\mathcal{S}}, x^*_{\mathcal{S}})$ of RPSP$_{\mathcal{S}}$, which is augmented with zeros to obtain the optimal solution of OP. The convergence checking is highlighted in Algorithm 2. The augmented RRMP$_{\mathcal{T}}$ contains fewer cuts (Pareto-optimal primal Cuts). Thus, its solving should be quick. To accelerate RRMP$_{\mathcal{T}}$ solving further, we can rely on a warm start using the last solution obtained augmented with zeros corresponding to the complicating variables in $\bar{\mathcal{T}}$.

To conclude this section, we make the following observation.

Observation 6. The number of iterations might be large if n is large. Still, it is worth highlighting that, since the solution procedure (column generation) of DWD converges by adding only a small number of the columns to the DWD RMP, it might be the same (BD is the dual of DWD) behavior for $RPSP_{\mathcal{S}}$.

Algorithm 2: Convergence checking

```
1 Insert all remaining complicating variables in $\bar{\mathcal{T}} = J \setminus \mathcal{T}$ into RRMP$_{\mathcal{T}}$ using the previously computed lifting
    coefficients.
2 if $|UB - LB| < \epsilon$ then
3     Return $(y^*_{\mathcal{S}}, x^*_{\mathcal{S}})$ optimal solution of RPSP$_{\mathcal{S}}$
4 else
5     Continue via the framework in Figure 3 until convergence.
6 end
```

This completes the discussion on the Accelerated $\mathcal{PBD}$ convergence. In what follows, to avoid selecting many complicating variables with a negative reduced cost (without being in the optimal solution) and increasing significantly the size of the subproblem, we only select the most promising complicating variable at each iteration, i.e., the variable y_ϕ such that $\phi = \min_{j \in J \setminus S} \{\bar{f}_j : \bar{f}_j < 0\}$. To highlight the method's benefits, we present next an illustrative facility location example.

4 Example

To highlight the benefits of the proposed method, consider the following facility location problem (FLP) example. A formulation for the capacitated facility location problem as given in Wentges (1996) is:

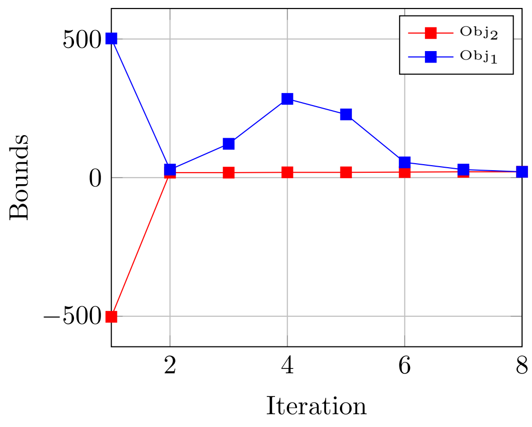
$$
\begin{aligned}
\mathbf{min} \quad & \sum_{j=1}^{n} f_j y_j + \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} && && \text{(FLP)} \\
s.t. : \quad & \sum_{j=1}^{n} x_{ij} = d_i && \forall i \in I \\
& x_{ij} \leq d_i y_j && \forall i \in I, j \in J \\
& \sum_{i=1}^{m} x_{ij} \leq s_j y_j && \forall j \in J \\
& x_{ij} \geq 0, \quad y_j \in \{0,1\} && \forall i \in I, j \in J
\end{aligned}
$$

where $n = |J| = 5$ (facilities), $m = |I| = 6$ (clients), $f_j = 1\, \forall j \in J$ (fixed cost), $s_j = 16\, \forall j \in J$ (capacity), $d = \{1, 1, 2, 3, 4, 5\}$ (demand), and unit cost matrix (variable cost)

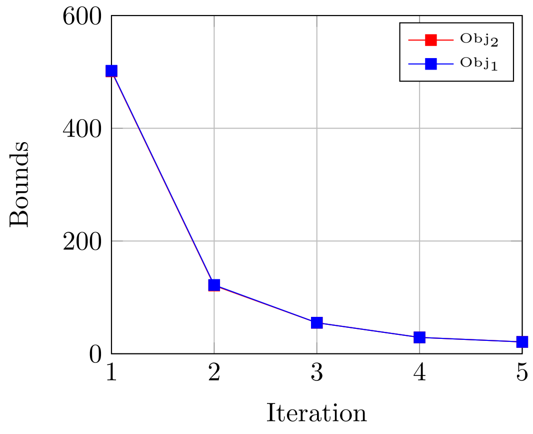
$$c = \begin{pmatrix} 60 & 10 & 20 & 30 & 1 \\ 10 & 1 & 20 & 30 & 40 \\ 40 & 20 & 30 & 1 & 20 \\ 40 & 20 & 1 & 10 & 20 \\ 1 & 20 & 40 & 40 & 40 \\ 1 & 20 & 40 & 40 & 40 \end{pmatrix}$$

The binary variable y_j is equal to 1 if facility $j \in J$ is opened and zero otherwise. The variable x_{ij} is the quantity supplied to customer $i \in I$ by facility $j \in J$. For this example, the optimal value is 21, and all 5 depots are opened. We compare the BD and the $\mathcal{PBD}$ using two scenarios. In the first scenario, $\bar{y} = (0, 0, 0, 0, 1)$ is provided as an initial point to BD. Equivalently, $\mathcal{S} = \{5\}$ is an initial point for $\mathcal{PBD}$. Figure 4 shows the results. For each method, we report Obj_1 (in blue) corresponding to the subproblem objective value and Obj_2 (in red) corresponding to the master problem objective value.

While BD requires 8 iterations (8 Benders cuts) to converge, $\mathcal{PBD}$ requires only 5 iterations (5 primal Benders cuts) to reach the optimal solution, i.e., the remaining 5 depots to open. Also, the objective value of the BD subproblem (Obj_1 in Blue on the left) shows a zigzagging behavior while the objective value of the $\mathcal{PBD}$ subproblem (Obj_1 in Blue on the right) shows a strict decrease. The necessity to close the gap between the UB and LB implies more iterations for BD. In contrast, $\mathcal{PBD}$ reaches optimality when no more depots with negative reduced costs are available, i.e., in five iterations.



(a) BD



(b) $\mathcal{PBD}$

Figure 4: Comparison between BD vs PBD for $\bar{y} = (0, 0, 0, 0, 1)$ and $\mathcal{S} = \{5\}$

Another interesting observation is the comparison between the Benders cuts generated. In the first iteration, the Benders cut generated using BD is:

$$z \geq 501 - 381y_2 - 219y_3 - 77y_4 - 78y_5 \tag{3}$$

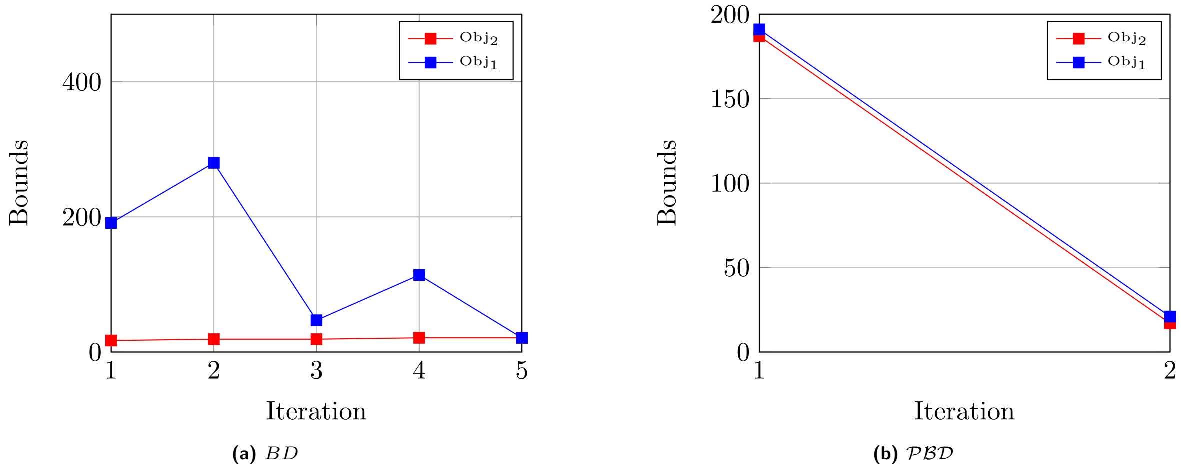
The Benders cut generated using $\mathcal{PBD}$ is:

$$z \geq 501 \tag{4}$$

Since y_1 is binary, the second cut dominates the first one as shown in Section 3.3.

In the second scenario, $\bar{y} = (0, 1, 1, 1, 1)$ is provided as an initial point to BD. Equivalently, $\mathcal{S} = \{2, 3, 4, 5\}$ is an initial point for $\mathcal{PBD}$. Figure 5 highlights the results for the second scenario. $\mathcal{PBD}$ reaches the optimal solution in two iterations while BD requires more. For $\mathcal{PBD}$, the first iteration corresponds to $\mathcal{S} = \{2, 3, 4, 5\}$ while the second iteration corresponds to $\mathcal{S} = \{1, 2, 3, 4, 5\}$. Two

Pareto-optimal cuts are required to reach the optimal solution. They are sufficient and provide all the necessary information. It is not the case with BD, which requires 5 Benders cuts to converge while showing a zigzagging behavior.



(a) BD (b) $\mathcal{PBD}$

Figure 5: Comparison between BD vs PBD for $\bar{y} = (0, 1, 1, 1, 1)$ **and** $\mathcal{S} = \{2, 3, 4, 5\}$

For this second example, the Benders cut generated using BD in the first iteration is:

$$z \geq 187 - 171y_1 \tag{5}$$

The Benders cut generated using $\mathcal{PBD}$ is:

$$z \geq 187 \tag{6}$$

Since y variables are binary, the second cut dominates the first one. A last observation is that all the Benders cuts generated by $\mathcal{PBD}$ are tight in the (relaxed) solution of RRMP$_{\mathcal{T}}$ and in each iteration. This is not the case for BD where a few cuts are tight in each iteration.

5 Experimental design

To study whether the $\mathcal{PBD}$ method is computationally efficient, we complement the theoretical analysis presented in previous sections with an extensive computational study. In this section, we describe the general characteristics of the test instances, the computational setting, and how the algorithms were implemented.

5.1 Instances

We test our method on the *facility location problem* (FLP), often used as a classical example in the BD context. We consider three different sets of instances from the literature. Here, we provide a high-level summary of these problems and instances. Further details can be found in the provided references.

The first instance set is the deterministic FLP used in Beasley (1988) and available in the OR Library. These benchmarks are probably the most widely used benchmarks when testing algorithm performance for the FLP. We have considered all 52 instances from the 16 classes available. Each class includes at most four instances with varying costs and capacity ratios. Also, these instances include up to 1000 customers with up to 100 potential facilities. A capacitated formulation of the deterministic case is FLP. We refer to these instances as *CAP* instances.

The second instance set is the stochastic variant of the facility location problem. For this variant, we have used the instances generated by Bodur et al. (2017) with up to 5000 scenarios. We have considered 80 instances from 4 classes and 5 scenarios. Each class includes four instances with varying costs and capacity ratios. Also, these instances include 50 customers with up to 25-50 potential facilities. A stochastic variant formulation is provided in Appendix A. We refer to these instances as $SCAP$ instances. The main reason behind the choice of stochastic instances is that BD, also referred to as the L-shaped method within the stochastic optimization community, is used significantly to tackle stochastic optimization problems. Thus, we aim to check the performance of the $\mathcal{PBD}$ in the stochastic case as well.

The third instance set is a large-scale version of the facility location problem used by Kratica et al. (2001) and available in the Max Planck Institut Informatik. These benchmarks are designed to be similar to real-life problems and have a large number of near-optimal solutions. We have considered all 22 instances from 6 classes. Each class includes at most five instances with varying costs and capacity ratios. Also, these instances have a similar number of customers and facilities with 2000 being the largest size. We refer to these instances as M instances.

5.2 Computational setting and implementation details

The coding language is C++ and tests are conducted using version 12.10.0 of IBM ILOG CPLEX solver. All experiments were carried out on a 3.20GHz Intel(R) Core(TM) i7-8700 processor, with 64GiB System memory, running on Oracle Linux Server release 7.7. We use real-time to measure runtime. In order to make the implementations simple and easily replicable, we do not use any specialized codes or algorithms. Thus, we solve all LPs and MILPs using IBM ILOG CPLEX 12.10.0 run on a single thread.

We compare the following four methods:

MILP: Solve directly with a MIP solver. In our case, we use the default CPLEX.

CPLEX BD: Solve with CPLEX implementation of BD. To do so, we set the CPLEX parameter *Benders Strategy* to option *Full*, i.e., CPLEX automatically decomposes the given model. We keep other CPLEX parameters to their default options. It is worth mentioning that the CPLEX-Benders is a state-of-the-art implementation of Benders decomposition (Baena et al., 2020).

BD: Solve with the standard BD as it was initially developed (Benders, 1962) without any acceleration strategies. BD is chosen as a baseline and allows comparing the number of Benders cuts.

PBD: Solve using the Accelerated $\mathcal{PBD}$ method.

There are several (meta)heuristics used to find an initial point for the FLP. While it is not our focus here, to find initial points, we implement the DROP heuristic that starts with all facilities open, keeps dropping (closing) the facility that gives the maximum decrease in the total cost, and stops if dropping any more facilities will no longer reduce the total cost (Nemhauser et al., 1978; Erlenkotter, 1978). Using this heuristic, we construct an initial point with as few open facilities as possible. For a fair comparison, we provide all methods with the same initial point. The time limit for solving any instance by any of the four methods is one hour.

6 Computational results

In this section, we quantify the computational benefits of the $\mathcal{PBD}$ when solving the instances considered. We first check the performance of the $\mathcal{PBD}$ on the CAP instances. Then, we evaluate its performance on the SCAP instances. After that, we highlight its performance on the M instances. We then quantify the impact of the implementation tips. Finally, we discuss some computational insights.

6.1 Deterministic Facility Location Problem

Table 1 presents the performance of the four methods on the CAP instances. We report the optimality gap (Gap) computed as $\frac{UB-OPT}{OPT}$, where OPT is the optimal value and UB is the best UB obtained, and the execution time ($Time$) in seconds. In the latter and when it is the case, we indicate between parentheses the number of instances that could not be solved to optimality. Furthermore, for the three decompositions, we report the number of Benders cuts ($Cuts$). Column $|J^*|$ refers to the number of opened facilities in the optimal solution. All the values are reported as averages.

Table 1: Performance Comparison for Cap Instances

CAP#	$\|I\|$	$\|J\|$	$\|J^*\|$	MILP		CPLEX BD			BD			PBD		
				Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time
41-44	50	16	9	0.00%	0.73	**4**	0.00%	**0.58**	24	0.00%	24.50	6	0.00%	0.62
51	50	16	8	0.00%	0.78	**4**	0.00%	**0.55**	34	0.00%	19.33	6	0.00%	0.59
61-64	50	16	8	0.00%	0.71	**4**	0.00%	**0.51**	32	0.00%	11.42	6	0.00%	0.52
71-74	50	16	7	0.00%	0.74	**5**	0.00%	**0.49**	28	0.00%	8.15	5	0.00%	0.51
81-84	50	25	11	0.00%	0.98	**8**	0.00%	**0.75**	>5000	4.11%	(4)	9	0.00%	0.77
91-94	50	25	11	0.00%	1.18	**8**	0.00%	**0.84**	>5000	3.93%	(4)	8	0.00%	0.85
101-104	50	25	10	0.00%	1.33	**5**	0.00%	**0.95**	48	3.78%	378.67 (1)	8	0.00%	0.96
111-114	50	50	11	0.00%	1.49	10	0.00%	**1.07**	>5000	4.27%	(4)	**9**	0.00%	1.12
121-124	50	50	11	0.00%	4.14	12	0.00%	**1.20**	>5000	4.77%	(4)	**8**	0.00%	1.26
131-134	50	50	10	0.00%	4.64	**7**	0.00%	**1.27**	>5000	3.97%	(4)	8	0.00%	1.36
a	1000	100	4	0.00%	232.00	7	0.00%	2.00	>5000	8.21%	(4)	**2**	0.00%	**1.71**
a1-4	1000	100	6	0.00%	72.50	15	0.00%	5.50	>5000	10.15%	(4)	**5**	0.00%	**3.57**
b	1000	100	7	0.00%	68.00	13	0.00%	2.55	>5000	8.34%	(4)	**5**	0.00%	**2.31**
b1-4	1000	100	9	0.00%	73.25	52	0.00%	13.25	>5000	12.33%	(4)	**7**	0.00%	**5.58**
c	1000	100	9	0.00%	104.00	15	0.00%	3.15	>5000	8.67%	(4)	**6**	0.00%	**2.48**
c1-4	1000	100	10	0.00%	101.25	43	0.00%	10.50	>5000	12.07%	(4)	**9**	0.00%	**5.86**
Avg	406	60	9	0.00%	41.73	13	0.00%	2.82	>5000	5.29%	2502.63	**7**	0.00%	**1.88**

From Table 1, we can infer that CPLEX BD and $\mathcal{PBD}$ outperform both MILP and BD. The latter fails to reach optimality on several CAP instances with the average gap being 5.29%. In many of them, BD generates more than 5000 Benders cuts without converging, highlighting the convergence issues of BD. Also, while CPLEX BD performs better compared to $\mathcal{PBD}$ on instances with 25-50 facilities and 50 customers, $\mathcal{PBD}$ outperforms other methods on larger instances with 100 facilities and 1000 customers. On average, $\mathcal{PBD}$ outperforms all other methods with an average time to optimality equal to 1.88 and an average number of Benders cuts equal to 7.

6.2 Stochastic Facility Location Problem

Table 2 presents the results obtained on SCAP instances. We consider from 250 to 5000 scenarios. The SCAP instances are more difficult than the CAP instances. Indeed, BD fails to reach optimality on all of them and MILP starts to fail from scenario 500. For CPLEX BD and $\mathcal{PBD}$, they both reach optimality in all instances. CPLEX BD requires many Benders cuts to reach optimality with the lowest number being 549 cuts and the highest number being 29037 cuts. On the other side, $\mathcal{PBD}$ requires far fewer cuts and converges in at most $|J^*|$ iterations. For execution time, $\mathcal{PBD}$ outperforms more significantly CPLEX BD on SCAP instances than CAP instances.

6.3 Large-scale Facility Location Problem

For the M instances, which mimic real-life cases, we report the results in Table 3. These instances are more complicated than CAP and SCAP instances since they have significantly more complicating variables. BD fails in these instances. MILP fails in instances with more than 500 customers and facilities. CPLEX BD also fails on instances with a size larger than 500 except for instance R2. $\mathcal{PBD}$ outperforms all methods and reaches optimality in all instances.

Table 2: Performance Comparison for SCAP Instances

K	SCAP#	$\lvert I\rvert$	$\lvert J\rvert$	$\lvert J^*\rvert$	MILP		CPLEX BD			BD			PBD		
					Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time
250	101-104	50	25	9	0.00%	18.00	601	0.00%	**4.00**	>5000	17.91%	(4)	**6**	0.00%	4.15
	111-114	50	50	16	0.00%	364.75	624	0.00%	13.00	>5000	9.62%	(4)	**13**	0.00%	**12.21**
	121-124	50	50	10	0.00%	557.00	894	0.00%	12.75	>5000	19.33%	(4)	**7**	0.00%	**12.14**
	131-134	50	50	9	0.00%	168.75	549	0.00%	7.00	>5000	23.33%	(4)	**7**	0.00%	**4.65**
	Avg	50	44	11	0.00%	341.91	667	0.00%	9.19	>5000	17.55%	-	**8**	0.00%	**8.05**
500	101-104	50	25	9	0.00%	64.75	1218	0.00%	**7.00**	>5000	19.23%	(4)	**6**	0.00%	8.75
	111-114	50	50	16	0.00%	1620.50	1736	0.00%	28.25	>5000	9.79%	(4)	**13**	0.00%	**27.58**
	121-124	50	50	10	0.24%	1988.25 (1)	1956	0.00%	27.25	>5000	19.63%	(4)	**7**	0.00%	**25.89**
	131-134	50	50	9	0.00%	760.5	1080	0.00%	14.00	>5000	23.61%	(4)	**7**	0.00%	**9.67**
	Avg	50	44	11	0.08%	1108.50	1497	0.00%	19.13	>5000	18.07%	-	**8**	0.00%	**17.97**
1500	101-104	50	25	9	0.00%	674.25	3575	0.00%	**27.75**	>5000	21.01%	(4)	**6**	0.00%	29.07
	111-114	50	50	16	7.69%	(4)	11044	0.00%	186.00	>5000	10.06%	(4)	**13**	0.00%	**137.77**
	121-124	50	50	10	11.60%	(4)	6838	0.00%	93.00	>5000	20.99%	(4)	**7**	0.00%	**89.57**
	131-134	50	50	9	3.60%	3259.25 (3)	3891	0.00%	52.25	>5000	26.54%	(4)	**7**	0.00%	**35.51**
	Avg	50	44	11	5.72%	2783.38	6337	0.00%	89.75	>5000	19.65%	-	**8**	0.00%	**72.98**
3000	101-104	50	25	9	4.31%	2407.75 (2)	5585	0.00%	62.25	>5000	23.11%	(4)	**6**	0.00%	**53.08**
	111-114	50	50	16	13.63%	(4)	29037	0.00%	309.00	>5000	11.07%	(4)	**13**	0.00%	**255.91**
	121-124	50	50	10	20.58%	(4)	10913	0.00%	208.00	>5000	23.09%	(4)	**7**	0.00%	**165.85**
	131-134	50	50	9	9.25%	(4)	7099	0.00%	126.50	>5000	29.19%	(4)	**7**	0.00%	**65.68**
	Avg	50	44	11	11.94%	3301.94	13158	0.00%	176.44	>5000	21.62%	-	**8**	0.00%	**135.13**
5000	101-104	50	25	9	7.41%	3106.25 (3)	11273	0.00%	112.50	>5000	25.42%	(4)	**6**	0.00%	**98.46**
	111-114	50	50	17	16.73%	(4)	22120	0.00%	528.25	>5000	12.17%	(4)	**13**	0.00%	**438.92**
	121-124	50	50	10	27.13%	(4)	16213	0.00%	380.75	>5000	25.40%	(4)	**7**	0.00%	**304.12**
	131-134	50	50	9	11.12%	(4)	11645	0.00%	216.75	>5000	32.11%	(4)	**7**	0.00%	**111.52**
	Avg	50	44	11	15.60%	3476.50	15313	0.00%	309.56	>5000	23.78%	-	**8**	0.00%	**238.26**

Table 3: Performance Comparison for M Instances

M#	$\lvert I\rvert = \lvert J\rvert$	$\lvert J^*\rvert$	MILP		CPLEX BD			BD			PBD		
			Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time
O1-5	100	4	**0.00%**	35.20	213	**0.00%**	2.33	>5000	5.33%	(5)	**4**	**0.00%**	**0.27**
P1-5	200	5	**0.00%**	139.00	1029	**0.00%**	43.80	>5000	6.26%	(5)	**4**	**0.00%**	**1.35**
Q1-5	300	5	**0.00%**	729.60	1766	**0.00%**	251.20	>5000	8.66%	(5)	**5**	**0.00%**	**4.77**
R1-5	500	6	13.01%	(5)	4075	2.64%	3229.80 (4)	>5000	12.12%	(5)	**6**	**0.00%**	**11.23**
S1	1000	6	42.11%	(1)	2562	2.92%	(1)	>5000	19.85%	(1)	**6**	**0.00%**	**27.59**
T1	2000	6	53.78%	(1)	627	1.96%	(1)	>5000	25.44%	(1)	**6**	**0.00%**	**174.86**
Avg	684	5	18.15%	1950.63	1712	1.25%	1787.86	>5000	12.94%	-	**5**	**0.00%**	**36.68**

For the M instances, $|J^*|$ is on average 5. Thus, $\mathcal{PBD}$ reaches the optimal solution in a few iterations despite the large size of facilities. This aligns with Observation 6 since the number of cuts is much less than the number of y variables and is rather of the order of the number of y variables in the optimal solution. On average, within 36.68 seconds, 5 Benders cuts are added to reach the optimal solution. In the remaining sections, we investigate further the methods and consider only the M instances with a large number of complicating variables (≥ 100).

6.4 Impact of implementation tips

In this section, we evaluate the impact of implementation tips on the $\mathcal{PBD}$. The abbreviations IP, WS, Cuts, and SS refer to the initial point, warm-start, Benders cuts, and selection strategy, respectively. In the first approach, we evaluate $\mathcal{PBD}$ when no initial point is provided. We recall that no initial point means the usage of an artificial solution with *high* costs as in DWD. In the second approach, we evaluate $\mathcal{PBD}$ when no warm start is applied to both RRMP and RPSP. In the third approach, we evaluate $\mathcal{PBD}$ when all integer optimality cuts are collected from the branch-and-bound tree of RPSP in a given iteration. We recall that we consider solely the local Pareto-optimal cut(s), obtained from the optimal node(s), in $\mathcal{PBD}$. Lastly, in the fourth approach, we compare $\mathcal{PBD}$ to $\mathcal{PBD}$ without the selection strategy. In the latter, we solve the RRMP to optimality and insert the support of its solution in the RPSP as described in Section 2. Table 4 highlights the effect of the implementation tips on the $\mathcal{PBD}$.

Table 4: Impact of acceleration strategies

M#	No IP			No WS			All Cuts			No SS			PBD		
	Cuts	Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time	Cuts	Gap	Time
O1-5	5	0.00%	0.53	4	0.00%	0.50	108	0.00%	0.61	4	0.00%	1.12	4	0.00%	0.27
P1-5	6	0.00%	2.24	4	0.00%	2.20	168	0.00%	2.65	4	0.00%	7.00	4	0.00%	1.35
Q1-5	6	0.00%	7.14	5	0.00%	7.12	245	0.00%	8.88	5	0.00%	22.80	5	0.00%	4.77
R1-5	7	0.00%	11.77	6	0.00%	18.40	571	0.00%	22.07	6	0.00%	110.80	6	0.00%	11.23
S1	7	0.00%	28.15	6	0.00%	48.30	1266	0.00%	57.96	6	0.00%	883.00	6	0.00%	27.59
T1	7	0.00%	184.32	6	0.00%	316.25	1872	0.00%	379.49	6	5.50%	(1)	6	0.00%	174.86
Avg	6	0.00%	40.19	5	0.00%	65.46	705	0.00%	78.61	5	0.92%	770.79	5	0.00%	36.68

On M instances, the initial point does not significantly impact the $\mathcal{PBD}$. This is due to the fact that $|J^*|$ is quite small. Thus, $\mathcal{PBD}$ reaches the optimal solution in at most 6 iterations. With a large $|J^*|$, the initial point will definitely accelerate the $\mathcal{PBD}$. Warm-starting significantly impacts the $\mathcal{PBD}$ performance, especially for large instances. For instance, while $\mathcal{PBD}$ reaches optimality in 174.86 seconds for the instance T1, $\mathcal{PBD}$ without warm-start requires 316.25 seconds. This is due to the large size of the subproblem and the time taken to solve it at each iteration from scratch. The same observation happens when all Benders cuts are inserted. The selection strategy also plays a crucial role, especially when J is large. Indeed, when solving the RMP to optimality and inserting the support of its solution in the RPSP$_{\mathcal{S}}$, several complicating variables that do not belong to the optimal solution might be inserted in the RPSP$_{\mathcal{S}}$. The latter becomes quite large and consequently computationally expensive. This is the case for instance T1, which can no longer be solved within one hour when the selection strategy is removed.

6.5 Computational insights

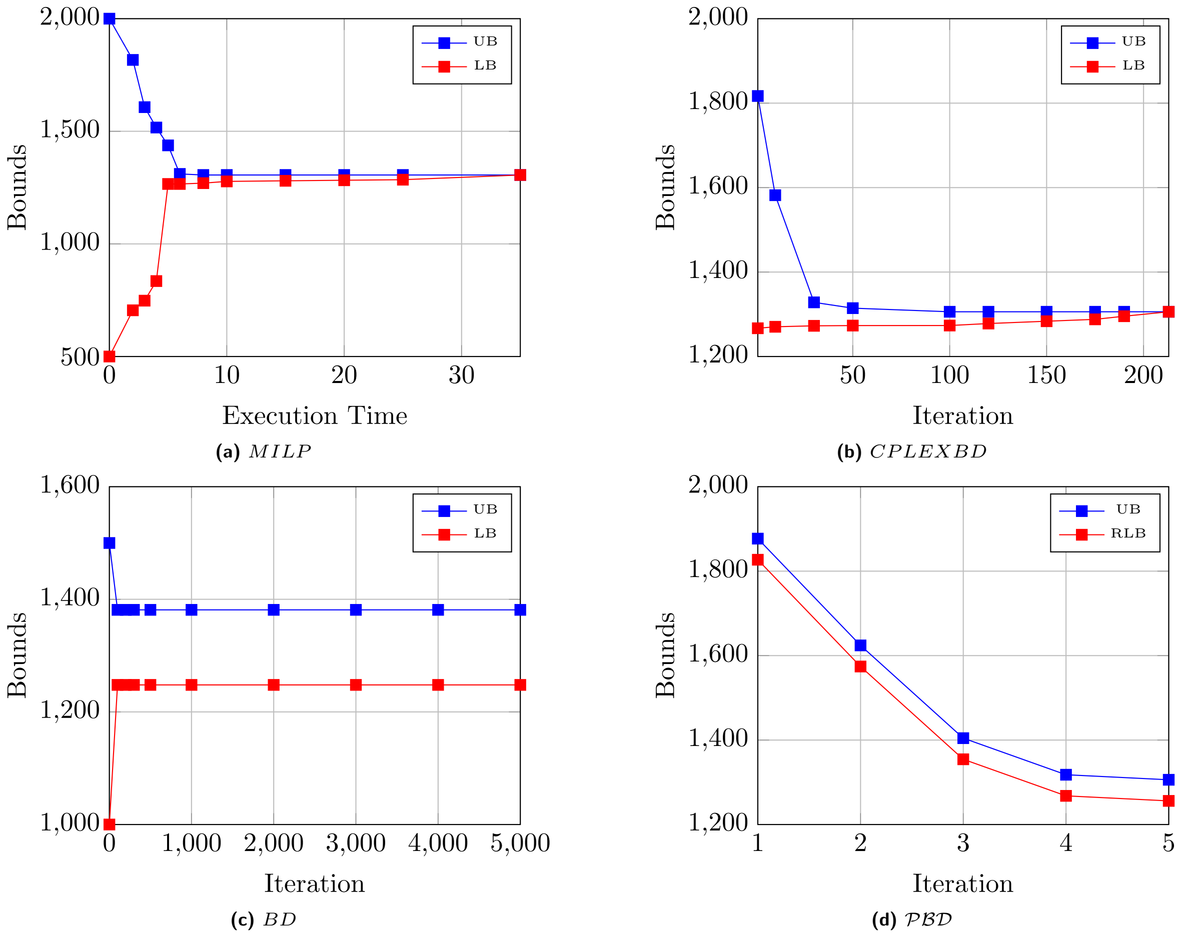
The results above report the optimality gap using the UBs obtained by each method. In what follows, we check the LBs as well and report in Table 5 the gap between the optimal solution and the LBs (Gap$^-$) within the time limit of one hour. It is computed as $\frac{OPT-LB}{OPT}$, where OPT is the optimal value and LB is the best LB obtained. The number of cuts and the execution time remain the same as in Table 3. For $\mathcal{PBD}$, we report the results before running Algorithm 2.

Table 5: Performance on the LBs

M#	$\lvert J^*\rvert$	MILP		CPLEX BD			BD			PBD		
		Gap$^-$	Time	Cuts	Gap$^-$	Time	Cuts	Gap$^-$	Time	Cuts	Gap$^-$	Time
O1-5	4	**0.00%**	35.20	213	**0.00%**	2.33	>5000	2.32%	(5)	**4**	3.83%	**0.27**
P1-5	5	**0.00%**	139.00	1029	**0.00%**	43.80	>5000	2.72%	(5)	**4**	2.10%	**1.35**
Q1-5	5	**0.00%**	729.60	1766	**0.00%**	251.20	>5000	3.76%	(5)	**5**	1.41%	**4.77**
R1-5	6	14.95%	(5)	4075	3.03%	3229.80 (4)	>5000	5.26%	(5)	**6**	**0.49%**	**11.23**
S1	6	48.39%	(1)	2562	5.65%	(1)	>5000	7.63%	(1)	**6**	**0.64%**	**27.59**
T1	6	61.80%	(1)	627	9.15%	(1)	>5000	10.06%	(1)	**6**	**1.89%**	**174.86**
Avg	5	20.86%	1950.63	1712	2.97%	1787.86	>5000	6.35%	-	**5**	**2.07%**	**36.68**

Table 5 highlights that $\mathcal{PBD}$ reaches the optimal solution without necessarily closing the gap between the UB and the RLB. While BD improves the LB more efficiently than the UB, both MILP and CPLEX BD improve the LB less efficiently than the UB. The performance of MILP, CPLEX BD, and BD is curbed by the necessity of closing the gap between the UB and the LB. In contrast, $\mathcal{PBD}$ does not have such an issue and can be stopped once there are no more promising y variables to insert, thus ensuring the practical superiority of the proposed method.

Figure 6 shows the behavior of the four methods on instance MO1. For MILP, we report the execution time along the x-axis. For CPLEX BD, BD, and $\mathcal{PBD}$, we report the iterations on the x-axis. The CPLEX BD requires 225 Benders iterations to converge. The BD does not converge and stagnates while the $\mathcal{PBD}$ reaches optimality in less than one second. The graphs confirm that MILP and CPLEX BD reach optimality on the UB in 6 and 0.75 seconds, respectively. Still, they do not converge because the LB is not tight. On both top graphs in Figure 6, much of the time is spent bringing the LB to optimality. On the bottom graphs, BD stagnates since both the UB and the LB do not change. On the bottom right side, $\mathcal{PBD}$ ensures a strict improvement from one iteration to another until reaching the optimal solution. This feature is one of the main strengths of $\mathcal{PBD}$.



(a) $MILP$ **(b)** $CPLEXBD$ **(c)** BD **(d)** $\mathcal{PBD}$

Figure 6: Comparison between the four methods for instance MO1

An interesting aspect to check is the number of Benders cuts tight at each iteration, i.e., when solving the RRMP$_{\mathcal{T}}$. Table 6 highlights such a comparison between BD and $\mathcal{PBD}$. We report the number of cuts as well as the average percentage of cuts tight (%Tight) in the RRMP$_{\mathcal{T}}$ solution. For BD, less than half of the cuts are tight at each iteration. For $\mathcal{PBD}$, all cuts added are tight at each iteration. It highlights the significant difference in the quality of Benders cuts added by each method. $\mathcal{PBD}$ provides the best cut(s) given a subset of y variables compared to BD. Thus, it implies a quicker convergence to the optimal solution.

Table 6: Tight cuts between BD and $\mathcal{PBD}$

M#	$\|J^*\|$	BD		PBD	
		Cuts	%Tight	Cuts	%Tight
O1-5	4	>5000	50.12%	**4**	**100.00%**
P1-5	5	>5000	47.74%	**4**	**100.00%**
Q1-5	5	>5000	45.93%	**5**	**100.00%**
R1-5	6	>5000	43.36%	**6**	**100.00%**
S1	6	>5000	40.12%	**6**	**100.00%**
T1	6	>5000	38.75%	**6**	**100.00%**
Avg	5	>5000	44.37%	**5**	**100.00%**

Another interesting point is that while BD relies on the gap between the UB provided by the subproblem and the LB provided by the master problem, $\mathcal{PBD}$ relies solely on the UB of the subproblem. Thus, instead of the double effort required to get both bounds close to each other in BD, a single effort is required to get $\mathcal{PBD}$ to reach the optimal solution. Once no more complicating variables are identified, $\mathcal{PBD}$ stops. Then, Algorithm 2 allows confirming that the optimal solution of the original problem is the optimal solution of the subproblem augmented with zeros. The master problem in $\mathcal{PBD}$ acts like a guide, i.e., it guides the subproblem toward the optimal solution. Indeed, as it is shown on the bottom right graph of Figure 6, the master and subproblem bounds behave in a similar and, interestingly, decrease strictly. From the computational insights above, we formulate the following conjecture.

Conjecture 1. For the FLP, the $\mathcal{PBD}$ converges strictly to the optimal solution in at most n iterations where n is the size of complicating variables.

To sum up the computational results, through the $\mathcal{PBD}$, we highlight the importance of leveraging the primal information when tackling optimization problems, especially when relying on decomposition techniques such as BD for large-scale problems. Indeed, using the primal information, one can follow improving directions towards the optimal solution. This aspect is lacking in all dual methods, including BD in its standard form. Furthermore, while we use the FLP to evaluate the $\mathcal{PBD}$, it is worth mentioning that it can be applied to many problems in the literature such as the multicommodity capacitated fixed charge network design problem, the stochastic network interdiction problem, the applications introduced in the Section 1, and many others (Adulyasak et al., 2015; Er Raqabi et al., 2023a; Himmich et al., 2023). Indeed, as highlighted with the M instances, it is very promising to apply it to real-life problems where the ratio $\frac{|J^*|}{|J|}$ is *small*.

7 Conclusion

This paper presents an attempt to view BD from a different perspective. Indeed, when seeing it as the dual of DWD, it is possible to leverage the primal information to converge more quickly compared to all dual BD methods. As far as we know, there has not been any such work in the literature. We believe that this paper will open a new era in BD with many future research directions. Several improvements can be made in the generation of cuts, the identification of promising complicating variables to insert, etc. The main strengths of the $\mathcal{PBD}$ are the generation of solely optimality cuts, the convergence to optimal or near-optimal solutions in at most the size of complicating variables, the improvement of the UB at each iteration, the handling of the integrality of the subproblem (a crucial issue in BD literature), and the scalability to several real-life problems since it relies on an intuitive dynamic insertion of complicating variables.

A The Stochastic Capacitated Facility Location Problem

Let us consider a set of potential facility locations J and a set of customers I. The objective is to open enough facilities to satisfy the customers with minimum cost. A customer must be served by at least one facility (or more). At each potential facility location $j \in J$, at most one facility with a service capacity of s_j can be opened. The corresponding fixed cost is f_j units. The variable service cost from facility $j \in J$ to customer $i \in I$ is c_{ij}. Each customer $i \in I$ has a stochastic demand d_i^k, where $k \in K$ is a scenario with probability p_k such that $\sum_{k \in K} p_k = 1$.

Let y_j be the binary variable equal to one if a facility is opened at location $j \in J$ and zero otherwise. Let $x_{ij}^k \geq 0$ be the quantity supplied from facility $j \in J$ to customer $i \in I$ given scenario $k \in K$. The stochastic capacitated facility location problem is then:

$$\begin{aligned}
\mathbf{min} \quad & \sum_{j \in J} f_j y_j + \sum_{k \in K} \sum_{j \in J} \sum_{i \in I} p_k c_{ij} x_{ij}^k & & \text{(SFLP)} \\
s.t.: \quad & \sum_{j \in J} x_{ij}^k \geq d_i^k & \forall i \in I, k \in K & \\
& \sum_{i \in I} x_{ij}^k \leq s_j y_j & \forall j \in J, k \in K & \\
& \sum_{j \in J} s_j y_j \geq \max_{k \in K} \sum_{i \in I} d_i^k & & \\
& x_{ij}^k \geq 0, \quad y_j \in \{0,1\} & \forall i \in I, j \in J, k \in K &
\end{aligned}$$

The objective minimizes the total costs (fixed and variable). The first constraint set controls the demand satisfaction for each customer under every scenario. The second constraint set controls the capacity level for each facility. The third constraint corresponds to the complete recourse property to the problem. The last constraints are positivity (x variables) and binary (y variables) conditions.

References

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2015). Benders decomposition for production routing under demand uncertainty. Operations Research, 63(4):851–867.

Baena, D., Castro, J., and Frangioni, A. (2020). Stabilized Benders methods for large-scale combinatorial optimization, with application to data privacy. Management Science, 66(7):3051–3068.

Beasley, J. E. (1988). An algorithm for solving large capacitated warehouse location problems. European Journal of Operational Research, 33(3):314–325.

Ben Amor, H., Desrosiers, J., and Valério de Carvalho, J. M. (2006). Dual-optimal inequalities for stabilized column generation. Operations Research, 54(3):454–463.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. Numerische mathematik, 4(1):238–252.

Bloom, J. A. (1983). Solving an electricity generating capacity expansion planning problem by generalized Benders' decomposition. Operations Research, 31(1):84–100.

Bodur, M., Dash, S., Günlük, O., and Luedtke, J. (2017). Strengthened Benders cuts for stochastic integer programs with continuous recourse. INFORMS Journal on Computing, 29(1):77–91.

Cai, X., McKinney, D. C., Lasdon, L. S., and Watkins Jr, D. W. (2001). Solving large nonconvex water resources management models using generalized Benders decomposition. Operations Research, 49(2):235–245.

Canto, S. P. (2008). Application of Benders' decomposition to power plant preventive maintenance scheduling. European journal of operational research, 184(2):759–777.

Codato, G. and Fischetti, M. (2006). Combinatorial Benders' cuts for mixed-integer linear programming. Operations Research, 54(4):756–766.

Contreras, I., Cordeau, J.-F., and Laporte, G. (2011). Benders decomposition for large-scale uncapacitated hub location. Operations research, 59(6):1477–1490.

Cordeau, J.-F., Soumis, F., and Desrosiers, J. (2001a). Simultaneous assignment of locomotives and cars to passenger trains. Operations research, 49(4):531–548.

Cordeau, J.-F., Stojković, G., Soumis, F., and Desrosiers, J. (2001b). Benders decomposition for simultaneous aircraft routing and crew scheduling. Transportation science, 35(4):375–388.

Er Raqabi, E. M., Himmich, I., El Hachemi, N., El Hallaoui, I., and Soumis, F. (2023a). Incremental LNS framework for integrated production, inventory, and vessel scheduling: Application to a global supply chain. Omega, 116:102821.

Er Raqabi, E. M., Wu, Y., El Hallaoui, I., and Soumis, F. (2023b). Towards resilience: Primal large-scale re-optimization. Les Cahiers du GERAD, G–2023–28.

Erlenkotter, D. (1978). A dual-based procedure for uncapacitated facility location. Operations Research, 26(6):992–1009.

Fischetti, M., Salvagnin, D., and Zanette, A. (2010). A note on the selection of Benders' cuts. Mathematical Programming, 124(1):175–182.

Geoffrion, A. M. (1970). Elements of large-scale mathematical programming part i: Concepts. Management Science, 16(11):652–675.

Guignard-Spielberg, M. and Spielberg, K. (2005). Integer Programming: State of the art and recent advances. Springer.

Himmich, I., Er Raqabi, E. M., El Hachemi, N., El Hallaoui, I., Metrane, A., and Soumis, F. (2023). MPILS: An automatic tuner for MILP solvers. Computers & Operations Research, 159:106344.

Jünger, M., Liebling, T. M., Naddef, D., Nemhauser, G. L., Pulleyblank, W. R., Reinelt, G., Rinaldi, G., and Wolsey, L. A. (2009). 50 Years of Integer Programming 1958-2008: From the early years to the state-of-the-art. Springer Science & Business Media.

Kratica, J., Tošic, D., Filipović, V., and Ljubić, I. (2001). Solving the simple plant location problem by genetic algorithm. RAIRO-Operations Research, 35(1):127–142.

Laporte, G., Louveaux, F. V., and Mercure, H. (1994). A priori optimization of the probabilistic traveling salesman problem. Operations research, 42(3):543–549.

Lee, J. (2008). A celebration of 50 years of integer programming. Optima, 76:10–14.

Lin, S., Lim, G. J., and Bard, J. F. (2016). Benders decomposition and an IP-based heuristic for selecting IMRT treatment beam angles. European Journal of Operational Research, 251(3):715–726.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. Operations research, 29(3):464–484.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions—i. Mathematical programming, 14(1):265–294.

Poojari, C. A. and Beasley, J. E. (2009). Improving Benders decomposition using a genetic algorithm. European Journal of Operational Research, 199(1):89–97.

Rahmaniani, R., Ahmed, S., Crainic, T. G., Gendreau, M., and Rei, W. (2020). The benders dual decomposition method. Operations Research, 68(3):878–895.

Rahmaniani, R., Crainic, T. G., Gendreau, M., and Rei, W. (2017). The Benders decomposition algorithm: A literature review. European Journal of Operational Research, 259(3):801–817.

Sherali, H. D. and Lunday, B. J. (2013). On generating maximal nondominated Benders cuts. Annals of Operations Research, 210(1):57–72.

Valério de Carvalho, J. M. (2005). Using extra dual cuts to accelerate column generation. INFORMS Journal on Computing, 17(2):175–182.

Wentges, P. (1996). Accelerating Benders' decomposition for the capacitated facility location problem. Mathematical Methods of Operations Research, 44(2):267–290.

Zeighami, V. and Soumis, F. (2019). Combining Benders' decomposition and column generation for integrated crew pairing and personalized crew assignment problems. Transportation Science, 53(5):1479–1499.