Chapter 4
Scenario decomposition framework for continuous second-stage problem: stochastic Benders decomposition

Cutting-plane schemes has been successfully used in solving both large-scale deterministic and stochastic problems since the pioneering paper of Geoffrion and Graves (1974), e.g., the uncapacitated network design problem with undirected arcs (Magnanti et al., 1986), stochastic transportation-location problems (Franca and Luna, 1982), locomotive and car assignment problems (Cordeau et al., 2000), stochastic supply chain design problems(Santoso et al., 2005; Uster and Agrahari, 2011), stochastic scheduling and planning of process systems (Saharidis et al., 2011; Yang and Lee, 2011) and the stochastic unit commitment problem (Peng and Jirutitijaroen, 2010), to name a few.

The combination of Benders algorithm principles and stochastic problems is commonly referred to as the stochastic Benders decomposition, or also commonly referred to as the L-Shaped Method (Van Slyke and Wets, 1969). In this context, the decomposition is carried out by decomposing the complete deterministic equivalent problem (Birge and Louveaux, 1997) into a Master Problem (MP), which comprises the complicating variables and related constraints, and a Slave Problem (SP), which is represented by the recourse decisions.

Nevertheless, under certain conditions, the traditional Benders decomposition (and consequently its stochastic version) might fail to achieve the aforementioned efficiency, a fact that has been broadly mentioned in the literature (see, for example Rei et al. (2007); Saharidis et al. (2010)). To circumvent this drawback various strategies have been proposed for accelerating Benders decomposition. McDaniel and Devine (1977) proposed the generation of cuts using the solution of a relaxed MP, and relaxing its integrality constraints. Furthermore, the authors present heuristic rules for determining when the integrality constraint is needed in order to ensure convergence of the algorithm. Although the results appear promising, the classical Benders decomposition can be more efficient in some cases. Cote and Laughton (1984) showed another approach for accelerating Benders algorithms. In their approach, the MP is

not solved to optimality but only the first integer solution obtained is used to generate the optimality or feasibility cut from the SP. The main drawback of this strategy is that by generating only cuts associated with suboptimal solutions, the algorithm may fail to generate cuts that are necessary to ensure convergence.

Within the context of generating more effective cuts, most researchers have sought either to generate additional "stronger" cuts at each iteration, or by modifying the way that Benders cuts are generated. Magnanti and Wong (1981) proposed a seminal procedure for generating Pareto-optimal cuts to strengthen the standard Benders optimality cuts, though with the often challenging requirement of identifying and updating a core point, which is required to lie inside the relative interior of the convex hull of the problem subregion defined in terms of MP variables. Papadakos (2008) highlights that the Magnanti-Wong's cut generation problem dependency on the solution of SP can sometimes jeopardize the algorithm's performance. To circumvent this difficulty, the author showed that one can obtain an independent formulation of the Magnanti-Wong cut generation problem. The author also provided guidelines for efficiently generating additional core points through convex combinations of previously known cores points and feasible solutions of the MP. More recently, Sherali and Lunday (2011) presented a different strategy for generating non-dominated cuts through the use of small perturbation on the right-hand-side of the SP to generate maximal non-dominated Benders cuts. The authors also showed a strategy based on complimentary slackness that simplifies the cut generation when compared with the traditional strategy used by Magnanti and Wong (1981).

Saharidis and Ierapetritou (2010) proposed the generation of an additional valid Benders cut based on a maximum feasible subsystem whenever a Benders feasibility cut is generated. These cuts were shown to significantly accelerate the convergence for problems where the number of feasibility cuts generated is greater than the number of optimality cuts. Fischetti et al. (2010) presented an alternative scheme that combines the generation of Benders cuts when both optimality and feasibility cuts are required. They formulate a subproblem where the generated cut acts as optimality and feasibility cuts. Rei et al. (2007) investigate how local branching techniques can be used to accelerate Benders algorithm. The authors also showed how Benders feasibility cuts can be strengthened or replaced with local branching. Saharidis et al. (2010) examined two applications of a scheduling problem, for which they demonstrated the effectiveness of generating covering cut bundles to enhance Benders cuts.

In this chapter, we present a framework for solving the two-stage stochastic programming model for a supply chain investment planning problem applied to petroleum products based on stochastic Benders decomposition. We also present the development of acceleration techniques tailored for the proposed approach. The proposed techniques address two different aspects in terms of algorithmic acceleration, since they aim at generating stronger cuts for the Benders decomposition in the context of stochastic programming, and they apply techniques for improving the quality of solutions obtained during the algorithm execution.

4.1 Mathematical Model

In this section we present the mathematical model considered for the development of the decomposition framework. We consider hereafter a simplified version of the supply chain investment model presented in chapter 2. In this case, we only consider discrete capacity expansion and arc projects.

(a) Nomenclature

The nomenclature used in this model is as follows:

Indexes and sets	
$i, j \in \mathcal{N}$	Locations
$p \in \mathcal{P}$	Products
$t \in \mathcal{T}$	Time periods
$\xi \in \Omega$	Scenarios
Subsets	
$\mathcal{B} \subseteq \mathcal{N}$	Distribution bases
$\mathcal{S} \subseteq \mathcal{N}$	Suppliers
Parameters	
A^0_{ij}	Current arc capacity
A_{ij}	Additional arc capacity
C_{ijt}	Transportation cost
D^{ξ}_{jpt}	Demand
H_{jpt}	Inventory cost
K_{jp}	Throughput capacity multiplier
L_{jp}	Security level multiplier
M^0_{jp}	Current inventory capacity
M_{jp}	Additional inventory capacity
O_{jpt}	Supply
P^{ξ}	Scenario probability

S_{jpt}	Shortfall cost
W_{jpt}	Inventory investment cost
Y_{ijt}	Arc investment cost
Variables	
x^{ξ}_{ijpt}	Product flow
v^{ξ}_{jpt}	Inventory level
u^{ξ}_{jpt}	Unmet demand
w_{jpt}	Inventory investment decision
y_{ijt}	Arc investment decision

Table 4.1: Model Notation

(b) Model Formulation

The mathematical model for the optimization of the aforementioned problem can be stated as follows:

$$\min_{w,y\in\{0,1\}} \sum_{j,p,t} W_{jpt} w_{jpt} + \sum_{i,j,t} Y_{ijt} y_{ijt} + \mathcal{Q}(w,y) \tag{4.1}$$

s.t.:

$$\sum_{t} w_{jpt} \leq 1 \quad \forall j \in \mathcal{B}, p \in \mathcal{P} \tag{4.2}$$

$$\sum_{t} y_{ijt} \leq 1 \quad \forall i,j \in \mathcal{N} \tag{4.3}$$

where w_{jpt} represents the capacity expansion investment decisions at location j for product p and in period t and y_{ijt} on arc connecting locations i and j investment decisions in period t, $\mathcal{Q}(w,y) = \mathbb{E}_{\Omega}[Q(w,y,\xi)]$ represents the expectation evaluated over all $\xi \in \Omega$ possible realizations for the uncertain parameters of the second-stage problem, given an investment decision (w,y). Constraints 4.2 and 4.3 define that each investment can happens only once along the time horizon considered.

The second-stage problem $\mathcal{Q}(w,y)$ can be stated as shown in equations 4.4 to 4.10.

$$\min_{x,v,u\in\mathbb{R}^{+}} \sum_{\xi} P^{\xi} \left(\sum_{i,j,p,t} C_{ijt} x^{\xi}_{ijpt} + \sum_{j,p,t} H_{jpt} v^{\xi}_{jpt} + \sum_{j,p,t} S_{jpt} u^{\xi}_{jpt} \right) \tag{4.4}$$

s.t.:

$$\sum_{i} x^{\xi}_{ijpt} + v^{\xi}_{jpt-1} + u^{\xi}_{jpt} = \sum_{i} x^{\xi}_{jipt} + v^{\xi}_{jpt} + D^{\xi}_{jpt} \quad \forall j \in \mathcal{B}, p \in \mathcal{P}, t \in \mathcal{T}, \xi \in \Omega \tag{4.5}$$

$$\sum_{j} x^{\xi}_{ijpt} \leq O_{ipt} \quad \forall i \in \mathcal{S}, p \in \mathcal{P}, t \in \mathcal{T}, \xi \in \Omega \tag{4.6}$$

$$\sum_{p} x^{\xi}_{ijpt} \leq A^{0}_{ij} + A_{ij} \sum_{t' \leq t} y_{ijt'} \quad \forall i, j \in \mathcal{N}, t \in \mathcal{T}, \xi \in \Omega \tag{4.7}$$

$$v^{\xi}_{jpt} \leq M^{0}_{jp} + M_{jp} \sum_{t' \leq t} w_{jpt'} \quad \forall j \in \mathcal{B}, p \in \mathcal{P}, t \in \mathcal{T}, \xi \in \Omega \tag{4.8}$$

$$v^{\xi}_{jpt} \geq L_{jp} \left(M^{0}_{jp} + M_{jp} \sum_{t' \leq t} w_{jpt'} \right) \quad \forall j \in \mathcal{B}, p \in \mathcal{P}, t \in \mathcal{T}, \xi \in \Omega \tag{4.9}$$

$$\sum_{i} x^{\xi}_{ijpt} \leq K_{jp} \left(M^{0}_{jp} + M_{jp} \sum_{t' \leq t} w_{jpt'} \right) \quad \forall j \in \mathcal{B}, p \in \mathcal{P}, t \in \mathcal{T}, \xi \in \Omega \tag{4.10}$$

The objective function 4.4 represents freight costs between nodes, inventory costs, and cost of shortfall. Equation 4.5 comprises the material balance in distribution bases. Constraint 4.6 limits the supply availability at sources. Constraint 4.7 defines the arc capacities and the possibility of its expansion through the investment decisions. In a similar way, constraint 4.8 defines the storage capacities together with its expansion possibility. Constraint 4.9 defines minimum inventory levels, according to safety requirements. Constraint 4.10 sets the throughput limit for bases, defined by the product of the available storage capacity with the throughput capacity multiplier.

4.2 Stochastic Benders Decomposition

To illustrate the following technique, let us assume that we have our problem written in the following compact notation:

$$v = \min_{x} cx + \mathcal{Q}(x) \tag{4.11}$$

s.t.:

$$Ax \leq b \tag{4.12}$$

$$x \in \{0,1\}^{n} \tag{4.13}$$

where $\mathcal{Q}$ is given by

$$\mathcal{Q}(x) = \min_{y} \sum_{\xi} P^{\xi} q y^{\xi} \tag{4.14}$$

s.t.:

$$Tx + Wy^{\xi} \leq h^{\xi} \quad \forall \xi \in \Omega \tag{4.15}$$

$$y \geq 0 \tag{4.16}$$

where c is a n-dimensional vector, q is a p-dimensional vector, A is a $m \times n$ matrix, b is a m-dimensional vector, T and W are matrices of size $q \times n$ and $q \times p$, respectively, and h is m-dimensional vector. In our context, cx represents the investment costs (i.e., first-stage costs), while $\sum_{\xi} P^{\xi} q y^{\xi}$ represents the costs with freight, inventory, and shortfall. (i.e., second-stage costs). The set of constraints $Ax \leq b$ represents constraints 4.2 and 4.3, while $Tx + Wy^{\xi} \leq h^{\xi}$ represents constraints 4.5 to 4.10.

The model proposed in section 4.1 can be defined as an optimization model with first-stage integer variables and second-stage continuous variables. Such characteristics allow us to consider a scenario-wise decomposition framework based on Benders decomposition (Benders, 1962) applied to stochastic optimization, given the particular diagonal structure of that problem, where the first-stage variables arise as complicating in a sense that they are the only elements providing connection between each scenario subproblem (Van Slyke and Wets, 1969). Moreover, the model has complete recourse (Birge and Louveaux, 1997), that is, for any feasible first-stage solution, the second stage problem is always feasible. Note that this fact is convenient since it precludes the generation of feasibility cuts in order to ensure feasibility in the context of the stochastic Benders decomposition.

We start by noting that the MP can be equivalently reformulated as follows:

$$v = \min_{x} c^T x + M \tag{4.17}$$

s.t.:

$$Ax \leq b \tag{4.18}$$

$$M \geq \mathcal{Q}(x) = \mathbb{E}_{\Omega}[Q(x, \xi)] \tag{4.19}$$

$$x \in \{0, 1\}^n \tag{4.20}$$

This formulation allows one to distinguish an important issue. Inequality 4.19 cannot be used computationally as a constraint since it is not defined explicitly, but only implicitly by a number of optimization problems. The main idea of the proposed decomposition method is to relax this constraint and replace it by a number of cuts, which may be gradually added following an iterative solving process. These cuts, defined as supporting hyperplanes of the second-stage objective function, might eventually provide a good estimation for the value of $\mathcal{Q}(x)$ in a finite number of iterations. In order to define the form of these cuts, let us first state the dual formulation of the second-stage problem, which represents the SP in our context, given a first-stage solution $\overline{x}$ and a

fixed scenario ξ:

$$Q(\overline{x}, \xi) = \max_{\pi} \pi^{\xi T}(h^{\xi} - T\overline{x}) \tag{4.21}$$

s.t.:

$$W^T \pi^{\xi} \leq q \tag{4.22}$$

$$\pi^{\xi} \geq 0 \tag{4.23}$$

Let Π denote the set of all extreme points of the polyhedron defined by the feasible space of the Dual Slave Problem (DSP) given by 4.22 and 4.23, k an element from Π, π denote the dual variables associated with constraint 4.15, and M the objective function value. Also, letting M^* be the optimal value, we must have $M^* \geq M^{(k)}, \forall k \in \Pi$. Therefore, our DSP can be restated as $\mathcal{Q}(\overline{x}) = \min_{M \geq 0}\{M : M \geq M^{(k)}, \forall k \in \Pi\}$, where

$$M^{(k)} = \sum_{\xi} P^{\xi} \pi_{(k)}^{\xi T}(h^{\xi} - Tx) = \pi_{(k)}^{T}(h - Tx) \tag{4.24}$$

Using the above representation for the DSP that is based on the extreme points $k \in \Pi$ of its polyhedron, we can now replace equation 4.19 with the new reformulation 4.24 for $\mathbb{E}_{\Omega}[Q(x, \xi)]$ in the MP, providing the following:

$$v = \min_{x} cx + M \tag{4.25}$$

s.t.:

$$Ax \leq b \tag{4.26}$$

$$M \geq \pi_{(k)}^{T}(h - Tx) \quad \forall k \in \Pi \tag{4.27}$$

$$x \in \{0, 1\}^n \tag{4.28}$$

This reformulation has the drawback of comprising a very large number of constraints of type 4.27. Moreover, at the optimal solution, not all of the constraints in 4.27 will be active. Therefore, in the iterative Benders decomposition algorithm, one works with a relaxed version of MP by considering only a subset of 4.27 at each iteration. We denote this subset by Π' which includes the constraints 4.27 generated via solving the DSP in the previous iterations. This relaxed formulation of the MP (RMP) considering only the subset Π' of cuts 4.27 provides a lower bound to the optimal solution of the MP.

At a given iteration of the stochastic Benders decomposition, a RMP$^{(k)}$ is solved first to obtain the values of $(\overline{w}^{(k)}, \overline{y}^{(k)})$. Then, these values are used to solve DSP$^{(k)}$ to obtain the values of dual variables $\pi^{(k)}$ (i.e., an extreme point $k \in \Pi$ of the dual polyhedron) and a new cut of the form 4.23 to include

into Π'. Note that when the $\text{DSP}^{(k)}$ is solved for given $(\overline{x}^{(k)})$, an upper bound for MP can be easily calculated by adding $\text{DSP}^{(k)}$'s objective value and the total fixed cost component for the $\text{RMP}^{(k)}$ (i.e., the objective value of $\text{RMP}^{(k)}$ excluding $M^{(k)}$).

4.3 Accelerating Benders Decomposition

In this section we present the techniques that we have developed for speeding-up the decomposition framework presented in the previous section. Figure 4.1 gives a schematic representation of the proposed algorithm, which is comprised by a combination of the traditional Benders decomposition framework and additional acceleration techniques that will be described in the following sections. The steps that represent acceleration ideas are represented in Figure 4.1 inside dashed boxes. The algorithm starts at an initialization

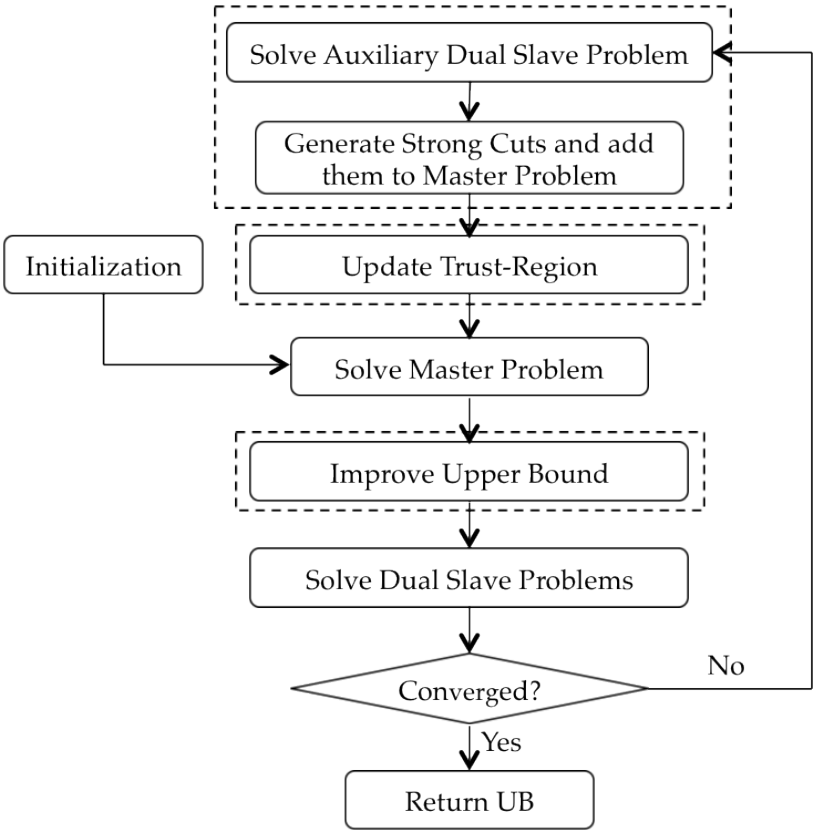

Figure 4.1: Schematic representation of the proposed stochastic Benders decomposition

step, where initial values for the parameters are set. The algorithm then proceeds with the iterative solution of the Master Problem (MP), followed by the procedure that seeks to improve the solution from it. The algorithm then proceeds to the solution of the DSP using either the solution from the MP

or the solution obtained in upper bounding improving procedure, depending on which is the best in terms of the value of the upper bound. This iterative procedure continues until the solution from the DSP and the MP are close enough, as measured by some stoping criteria. If the criteria are not fulfilled yet, the procedure iterates solving an auxiliary DSP to derive the strong cuts that are added in the sequence to the MP and, which is solved again in a new iteration. Notice that, even though the algorithm follows a classical iterative framework, it has particular features that differ from traditional approaches.

(a) Multi cut framework

Recall that in the stochastic Benders decomposition presented in the previous section, a single optimality cut is added at each iteration. This cut aims at approximating the value of the second-stage function at the current solution. However, instead of using only a single cut at each iteration, one can add multiple cuts to approximate the individual second-stage cost function corresponding to each one of the $|\Omega|$ scenarios. In this case, the RMP can be reformulated as follows:

$$\min_x c^T x + \sum_{\xi} P^{\xi} M^{\xi} \tag{4.29}$$

s.t.:

$$Ax \leq b \tag{4.30}$$

$$M^{\xi} \geq \pi_{(k)}^{\xi T}(h^{\xi} - Tx) \quad \forall \xi \in \Omega, k \in \Pi \tag{4.31}$$

$$x \in \{0,1\}^n \tag{4.32}$$

Birge and Louveaux (1988) showed that the use of such a framework may greatly speed-up convergence. The main idea behind this multi cut framework is to generate an outer linearization for all functions $Q(x,\xi)$, replacing the outer linearization of $\mathcal{Q}(x)$. The multi cut approach relies on the idea that using independent outer approximations of all functions $Q(x,\xi)$ provide more information to the MP than the single cut on $\mathcal{Q}(x)$, and therefore, fewer iterations are needed to reach the optimal solution.

(b) Generating stronger cuts

Magnanti and Wong (1981) proposed a seminal methodology to accelerate convergence of Benders decomposition by strengthening the generated cuts. They observed that in certain cases where the SP presents degeneracy, one might generate different cuts for the same optimal solution ($\overline{x}^{(k)}$), each

one of different strength in terms of efficiently approximating the second-stage cost function. To circumvent this difficulty, the authors proposed a methodology for identifying the strongest possible cut, which they referred to as the Pareto-optimal cut. Figure 4.2 illustrates this fact, showing distinct possibilities for cut generation, provided a solution $x^{(k)}$, where $c^{\star}(x^{(k)})$ represents the strongest cut in this context. Similarly to what we did in section 4.2, let

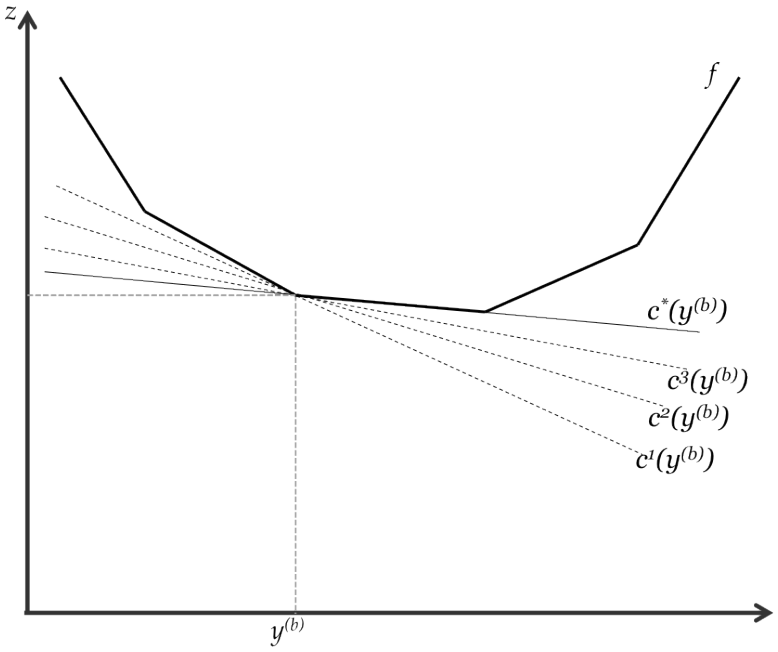

Figure 4.2: Geometric illustration of cut strength

$\Pi = \{\pi \in \mathbb{R}^q | W^T\pi \leq q\}$ be the set of feasible solutions for DSP. In addition, let $M \geq \pi^T (h - Tx)$ represent a Benders cut, and Π_{alt} be the set of alternative optimal solutions for the DSP given a MP solution $\overline{x}$. Then, we say that the Benders cut generated by π^* dominates all others (i.e., is Pareto-optimal) if:

$$\pi^{*T}(h - T\overline{x}) \leq \pi^T(h - T\overline{x}) \quad \forall \pi \in \Pi_{alt} \tag{4.33}$$

Magnanti and Wong (1981) showed how one can generate Pareto-optimal cuts based on the notion of core points. A core point is defined as a point $\hat{x}$ in the relative interior of $Conv(X)$, where $Conv(\cdot)$ denotes the convex hull (Wolsey, 1998). They proved that if a cut is selected such that it attains the maximum value at a core point amongst the set of all alternative cuts, then this cut is not dominated by other cuts at any feasible solution - a Pareto-optimal cut.

In order to generate these cuts, they suggest selecting some core point $\hat{x}$, and after solving SP for $\overline{x}$ (which we denote hereafter as SP($\overline{x}$)), they generate the Benders cut by subsequently solving a secondary subproblem, which can

be stated as follows:

$$\max\{\pi^T(h - T\hat{x}) \mid \pi \in \Pi_{alt}\} \tag{4.34}$$

where $\Pi_{alt} = \{\pi \in \Pi \mid \pi^T(h - T\overline{x}) = f(SP(\overline{x}))\}$ is the set of alternative optimal solutions of 4.34 and $f(\cdot)$ the objective function value.

Nevertheless, there are certain implementation issues related to Magnanti and Wong's cut generation procedure. First, the dependency of the subproblem 4.34 to the DSP might jeopardize the algorithm efficiency, especially in the cases where the DSP might turn out to be difficult to solve. Moreover, since Magnanti and Wong's procedure requires a new core point at each major iteration (recall that the core points rely inside the convex hull of the MP, whose feasible region is changing at each major iteration), it might be the case that it is not easy to obtain new core points at each iteration. To address this drawback, researchers often approximate core points (Santoso et al., 2005; Papadakos, 2009), arbitrarily define then by fixing components of the core point vector (Mercier et al., 2005) or use alternative points derived from a given problem structure (Papadakos, 2008). In addition to that, it is important to note that this strategy does not always yield a net computational advantage since the trade-off between the reduction in the number of iterations required compared to the increase in the number of linear programs solved to generate each cut might not pay-off (Mercier and Soumis, 2007).

In this chapter we propose an alternative way of generating nondominated cuts based on the definition of maximal cuts. Following the ideas of Sherali and Lunday (2011) for generating maximal nondominated Benders cuts, we show an alternative for strengthening the Benders Cuts while circumventing the aforementioned drawbacks.

First, we start by highlighting the standard definition of maximal, typically used in cutting plane theory from integer programming literature (Wolsey, 1998). Let us rewrite the Benders cut generated from a selected $\pi \in \Pi_{alt}$ as:

$$M \geq \pi^T h + \sum_{j=1}^{n}(-\pi^T T_j)\overline{x}_j \tag{4.35}$$

Then, we say that a Benders cut is maximal for a given π if, for every $\pi' \in \Pi_{alt}$, we have that $\pi^T h \geq \pi'^T h$ and $-\pi^T T_j \geq -\pi'^T T_j$. It is not difficult to see that a Pareto-optimal or nondominated cut generated in the way proposed by Magnanti and Wong (1981) can be also considered maximal, provided that the core point $\hat{x}$ is positive.

Sherali and Lunday (2011) show that the aforementioned concept of maximal cuts can be used to derive an alternative way of generating cuts that would accelerate Benders decomposition. To achieve such a goal, we must first view the process of generating maximal cuts as one of determining a Pareto-optimal solution to the multiple objective problem defined as:

$$\max\{\pi^T h, -\pi^T T_1, \ldots, -\pi^T T_n \mid \pi \in \Pi_{alt}\} \tag{4.36}$$

We can obtain a solution for this problem by selecting a positive weight vector (PWV) and then maximizing the positive weighted sum of the multiple functions in 4.36 (Steuer, 1989). By doing this, we end up by having to solve the following problem:

$$\max\{\pi^T h + \sum_{j=1}^{n} -\pi^T T_j \hat{x} \mid \pi \in \Pi_{alt}\} \tag{4.37}$$

which is exactly the problem defined in 4.35. Therefore, if we define $\hat{x}$ as a positive core point solution, then the resulting cut would be both maximal as well as nondominated.

Seeking to obtain an efficient framework to derive these cuts, we can combine in the same problem both the step where we solve DSP($\overline{x}$) to obtain Π_{alt} and the subsequent step of solving 4.34. Toward this end, we must first note that we are essentially considering a priority multiple objective program, where we want to first maximize DSP($\overline{x}$) (i.e., maximize $\pi^T(h - T\overline{x})$ subject to $\pi \in \Pi$) and next, considering all alternative solutions to this problem, choose the one which maximizes $\pi^T(h - T\hat{x})$. Again, one might notice that the approach of Magnanti and Wong (1981) to generate nondominated cuts using the core point $\hat{x}$ can be interpreted in the same way.

Sherali and Soyster (1983) showed that such a multiple objective program can be equivalently solved by the following combined weighted sum problem:

$$\max\{\pi^T(h - T\overline{x}) + \mu[\pi^T(h - T\hat{x})] \mid \pi \in \Pi\} \tag{4.38}$$

where μ is a suitably small weight. Although Sherali and Soyster (1983) showed that it is always possible to derive μ such that it would render 4.38 equivalent to the multi-objective problem 4.37, the derivation of such a weight is not typically a practically convenient task except in some particular cases.

In order to circumvent this drawback, we propose an alternative way of dealing with the weight μ in order to obtain what we call as *dynamically updated near-maximal Benders cuts*. The main reasoning behind the following

ideas are rather experimental than theoretical. What we observe from our numerical experiments is that the solutions obtained in the early iterations yield poor descriptions of the second-stage cost curve, which is exactly what we are trying to approximate through the use of Benders cuts in the stochastic programming context. Moreover, we observe that by applying the aforementioned ideas of generating maximal cuts, we can consider 4.38 as an auxiliary problem to simulate the existence of more dense first-stage solutions in the early iterations in order to speed-up the convergence. As for the algorithmic procedure, we can then iteratively adjust the weight μ in order to favor solutions that are more focused on improving the original $DSP(\overline{x})$ objective value $\pi^T(h - T\overline{x})$ rather than 4.34.

One important characteristic of such a framework for updating the weight μ is that it does not prevent convergence if a proper sequence $\{\mu_{(k)}\}_{k=1,\ldots,\infty}$ is selected. In order to keep the original convergence properties of the traditional Benders decomposition, it follows that one might select a sequence of $\mu_{(k)},\ k = 1, \ldots, \infty$ such that the following properties hold:

1. $\sum_{k=1}^{\infty} \mu_{(k)} \to \infty$
2. $\mu_{(k)} \to 0$ as $k \to \infty$

By selecting such a divergent series, its is not difficult to see that convergence is guaranteed since:

$$\lim_{\mu \to 0} \pi^T(h - T\overline{x}) + \mu[\pi^T(h - T\hat{x})] = \pi^T(h - T\overline{x}) \tag{4.39}$$

allowing us to rely on the results from Benders (1962) (or from Van Slyke and Wets (1969) for the stochastic version, or even from Birge and Louveaux (1988) for the multi cut framework), which guarantees convergence for the algorithm.

(c) Additional acceleration ideas

Combined with the strengthening of the cuts generated at each major iteration of the proposed algorithm, we also use additional acceleration ideas in order to improve computational efficiency.

Upper bound improving

In our implementation of the stochastic Benders decomposition, we observed that there is a strong relationship between the quality of the incumbent solutions ($\overline{x}$) obtained during the execution and the convergence rate of the algorithm. This issue is related with the fact that, especially in the early iterations, the incumbent solutions obtained may be quite far from the optimal

solution, leading the algorithm to explore inferior parts of the feasible region. However, if a good solution ($\overline{x}$) is made available through the use of some heuristic, we can use it in place of the incumbent solution and proceed from there. Our algorithm makes use of a particular heuristic in order to try to generate these good solutions during the algorithm execution.

The heuristic relies on facts observed during our computational experiments. We observed that, after the optimality gap becomes reasonably small, the bounds exhibit a tailing off behavior as the iterations progress. This effect is mainly due to the fact that, in these iterations, all the incumbent solution tend to present identical or very similar selection of projects (in terms of location and product for locations and origin and destination for arcs), only changing timing decisions. Because the timing decisions have relatively small influence on the objective value, the upper bound changes very little. In order to avoid this behavior, this heuristic is applied after a certain number of iterations with no improvement on the upper bound. The heuristic consists of three main steps:

1. Fix the current project selection to those in the current incumbent solution.

2. Randomly sample a subset of the scenario set Ω and solve the equivalent deterministic to determine whether these investments should be selected indeed, and if so, when. Notice that by doing this, we are both reducing the size of the second-stage problem (since it is a subset of Ω), as well as the size of the first-stage problem (since we are only considering investments decided in terms of location and product for tankage projects and origin and destination for arc projects, hence considering fewer integer variables).

3. Evaluate the obtained solution to check if it provides an improved upper bound. If so, use this solution to update the incumbent solution and the correspondent upper bound as the incumbent upper bound.

Trust-region

As pointed out by Ruszczyński (1997), the initial iterations of decomposition methods based on cutting planes tend to present an unstable behavior. This effect is mainly due to the fact that the solutions tend to oscillate between different sections of the feasible region, what may lead to slow convergence.

In the continuous case, this effect can be effectively controlled by the use of two different approaches. The first consists of adding a regularizing term in

the Master Problem objective function that penalizes the l_2-distance between the current solution and the previous one (Ruszczyński, 1997). The second focuses on constraining the l_∞ distance of the MP variables from the previous solution within a trust region (Linderoth and Wright, 2003). These extensions prevent the MP solution from moving far from the previous iterate. One point that must be highlighted is that both the penalty magnitude and the size of the trust region must be controlled during the execution of the algorithm based on its progress. Using a proper control is imperative when using these techniques in order to avoid losing convergence properties.

In our problem, the first-stage variables are binary vectors. In this case, using a l_2 regularizing term would render a mixed-integer quadratic MP, which would become much more complex in terms of solution methodology. Moreover, a l_∞ trust-region would be useless in our case. Since feasible MP solutions are extreme points of the unit hypercube, a trust region of size greater or equal than one would include all its vertices (i.e., all possible binary feasible solutions), while a trust-region with size less than one would include only the previous solution.

Santoso et al. (2005) show how one can deal with this drawback by using the Hamming distances between the binary solution vector as a measuring unit for the trust region. Let $(\overline{x}^{(k)})$ be the MP solution at iteration k and let $\mathcal{X} = \{j = 1, \ldots, n \mid \overline{x}_j^{(k)} = 1\}$. Then, we impose the following constraint in the MP to be solved in iteration $k + 1$:

$$\sum_{j \in \mathcal{X}} (1 - x_j) + \sum_{j \notin \mathcal{X}} x_j \leq \Gamma^{(k+1)} \tag{4.40}$$

where $\Gamma^{(k+1)} < n$ represents the trust-region size in iteration $k + 1$. Unfortunately, convergence cannot be guaranteed if a non-redundant trust region is used throughout the algorithm execution. Hence, since the algorithm tends to have the oscillating effect that we are willing to avoid mostly in the beginning of the execution, we dynamically adjust the size as the algorithm converges. When the algorithm reaches a sufficiently small optimality gap, we remove 4.40 from the MP in order to ensure convergence.

(d) Algorithm statement

We can summarize the proposed algorithm as follows:

Step 1: Initialization:

1.1) Set UB $= \infty$; LB $= -\infty$; $k = 1$

Step 2: Solve Master Problem:

2.1) If $LB_k > LB$, then $LB = LB_K$ and $(\overline{x}^{(k)}) = (x^{(k)})$.

Step 3: Solve Dual Slave Problems:

3.1) Solve each subproblem $Q(\overline{x}, \xi), \xi = 1, \ldots, \Omega$ and combine then to obtain $UB = \mathbb{E}_\Omega[Q(\overline{x}, \xi)]$

Step 4: If the limit for the number of successive iterations without improvement on LB is reached, then execute the upper bound improving procedure

4.1) Apply the proposed heuristic for generating an alternative first-stage solution $(x^{(k)})_{alt}$;

4.2) Evaluate $(x^{(k)})_{alt}$ so that $UB_{alt} = \mathcal{Q}(\overline{x})^{(k)}$. If $(x^{(k)})_{alt}$ is better than $(x^{(k)})$ (i.e, $UB_{alt} < UB$, then make $(x^{(k)}) = (x^{(k)})_{alt}$

Step 5: If $UB - LB < \epsilon$ or any other criteria, such as time elapsed or number of iterations are met, stop and return $(x^{(k)})$ and UB. Otherwise, set $k = k + 1$ and proceed.

Step 6 Cut generation:

6.1) Update parameter $\mu^{(k)}$;

6.2) Solve the auxiliary Dual Slave Problem 4.38 to obtain $\pi^{(k)}$;

6.3). Generate strong cut as 4.35 and add it to the Master Problem;

Step 7: Update the trust-region constraint 4.40 in the Master Problem. Return to *Step 2*.

4.4 Numerical Experiments

This section describes the computational experiments performed to evaluate the proposed algorithm under different considerations. All experiments described in this section were executed in an Intel Xeon 2.4GHz CPU with 4GB RAM and implemented in AIMMS 3.12. The mixed-integer and linear programming models within the decomposition framework were solved with CPLEX 12.3.

In order to assess the efficiency of the proposed framework, we consider an instance which consists of the realistic case study of a large-scale investment planning problem described in section 3.3. We compare the results with three different techniques. The first technique used (for now on referred as *Algorithm 1*) generates nondominated Benders cuts according to Magnanti and Wong (1981), with the approximation and core point updating technique as proposed by Papadakos (2008), i.e., we initialize a core point approximation $\hat{x}$ with a feasible solution to the MP and then update the approximation at each successive iteration by setting $\hat{x} = \lambda\hat{x} + (1 - \lambda)\overline{x}$. We adopt $\lambda = 0.5$, as prescribed by Papadakos (2008). The author states that, based on empirical observation, such a value for λ usually yields better results in terms of

algorithmic convergence. The second technique (*Algorithm 2*) used consists of generating maximal nondominated Benders cut as proposed by Sherali and Lunday (2011). The authors show that one can use the following expression to calculate μ that yields a near-optimal maximal Benders cut:

$$\mu = \frac{\epsilon_0}{M\theta} \tag{4.41}$$

where ϵ_0 is a prespecified tolerance on the absolute optimality gap, M is the penalty for recourse unfeasibility (that is equivalent to the cost of unmet demand in our case), and $\theta = \epsilon_0 + \max\{0, max\{\hat{h}_i\}\} - \min\{0, \min\{\hat{h}_i\}\}$, with $\hat{h} = h - T\hat{x}$. We used a fixed value for the weight μ, as shown by the authors in their calculations. This parameter was empirically derived, since it is not practically convenient to base the selection of ϵ_0 on the optimal solution. Note that by doing that, we are in effect implicitly dictating a particular choice of ϵ_0. Finally, *Algorithm 3* use the *dynamically updated near maximal Benders cuts* we have proposed in section 4.3(b) as the cut generation strategy. In this case we update the weight according to the following divergent series:

$$\mu^{t+1} = \frac{g}{h|B|}\mu^t \tag{4.42}$$

where g and h are prespecified parameters, and $|B|$ represents the current size of the set of generated cuts. In both experiments we use $g = 2$, $h = 1$, and μ_0 is given by 4.41. We use ϵ_0 as an empirically fixed value. In all cases, the algorithms were developed considering the multi-cut version. All experiments in this example were solved up to 2% optimality gap.

In order to assess the efficiency of the proposed approach, we performed two different experiments. The first experiment seeks to illustrate the effects of using the acceleration techniques "Upper bounding improving" and "Trust regions" in a random sample composed by 200 scenarios. Table 4.2 summarizes the performance of these techniques compared them individually and in combination with the case where none of such acceleration techniques are used (column "Without"). Note that in Table 4.2 the bounds are not reported at their last iteration since we are using arbitrary increments in the number of iterations.

# Iter.	Without		UB Imp.		TR		UB Imp. + TR	
	UB	%gap	UB	%gap	UB	%gap	UB	%gap
1	1732310.5	99.2	1732310.5	99.2	1732310.5	99.2	1732310.5	99.2
5	481802.2	46.3	481802.2	46.4	481802.2	46.4	481802.2	46.3
10	418213.5	7.3	416152.5	10.1	416152.5	3.4	418213.5	7.3
15	412056.4	3.7	408188.4	2.5	411950.4	2.8	408188.4	2.4
20	412056.4	2.7	-	-	411950.4	2.3	-	-
25	412056.4	2.3	-	-	-	-	-	-
30	412056.4	2.1	-	-	-	-	-	-
CPU Time(s)		1151.2		617.7		619.0		446.3

Table 4.2: Summary of CPU times(s) - experiment 1

Table 4.2 allows us to observe the effect of the acceleration techniques "Upper bound improving" and "Trust regions" separately (indicated as "UB Imp." and "TR", respectively) and compare it with the case where none of these acceleration techniques are used (indicated as "No Acc."). From the results we can conclude that both techniques improve convergence of the proposed algorithm, reducing the total CPU time required to reach the convergence criterion by approximately 46% in both cases. Moreover, when we combine both techniques, the reduction in the solution times is even larger, yielding improvement on the solution time of over 61%.

For the second experiment, we developed a set consisting of 100 independent scenario samples of 10 different sizes varying from 20 to 200 scenarios, as can be seem in Table 4.3. Notice that in this case, we are solving the problem for 1000 different demand instances, since the samples are independent. Our objective by doing that is to assess the efficiency of the proposed approach independent of particularities of a given scenario sample.

Scenarios	*Algorithm 1*		*Algorithm 2*		*Algorithm 3*		CPLEX	
	Avg.	St. Dev.	Avg.	St. Dev.	Avg.	St. Dev.	Avg.	St. Dev.
20	56.5	19.2	55.5	15.0	22.3	2.2	9.6	0.6
40	104.7	37.7	102.1	31.3	43.2	2.7	33.7	3.3
60	162.4	66.5	140.7	52.0	64.0	4.5	51.5	5.0
80	219.5	78.6	221.5	111.8	83.7	5.8	104.0	20.9
100	287.2	124.3	272.5	139.2	109.1	12.7	185.6	34.5
120	369.6	150.5	313.9	152.4	128.3	9.2	287.8	46.3
140	387.6	163.3	381.8	170.9	151.2	10.8	473.0	122.3
160	483.8	208.8	372.3	135.8	176.7	11.6	601.1	140.8
180	612.7	270.3	477.1	189.4	203.8	13.9	734.9	165.6
200	631.4	252.7	521.8	196.1	230.1	12.2	927.6	161.0

Table 4.3: Summary of CPU times(s) - experiment 2

Table 4.3 presents the statistical data retrieved from the experiments carried out in the second experiment, showing the average time solution (Avg. column) and the standard deviation (Std. Dev. column) in CPU seconds. As can be seen in Table 4.3, the CPLEX times are smaller in the experiments when a small number of scenarios is considered. As the number of scenarios increases, the decomposition frameworks outperform the use of CPLEX. We also highlight the performance of our algorithm (*Algorithm 3*) in the case study under consideration. As we can be observe from the experimental results, the proposed algorithm performed better than the other cutting generation strategy in all experiments. In addition, for cases where more than 80 scenarios where considered, *Algorithm 3* reached the best average solution times among all solution procedures, including CPLEX.

Another remarkable feature that can be observed in Table 4.3 is related to the standard deviation of the solution times. The results suggest that *Algorithm 3* attains the smaller deviation in terms of CPU seconds regarding the time the algorithm takes to reach a 2% gap suboptimal solution. The observation of this fact lead us to the conclusion that the performance of our *Algorithm 3* is less affected by particular characteristics of the scenario sample itself since it seems to be more robust in terms of solution time variation.

4.5 Conclusions

In this chapter we have presented the development of acceleration techniques for the stochastic Benders decomposition to solve the investment planning problem applied to the petroleum products supply chain. We have pro-

posed a new methodology for generating *dynamically updated near-maximal Benders cuts*, and compared it with acceleration techniques proposed by Papadakos (2008) and Sherali and Lunday (2011) for the stochastic Benders algorithm. Moreover, we have proposed the application of two additional acceleration techniques to further improve the convergence of the algorithm, especially in cases where convergence is difficult due to the computational complexity of the problem at hand.

We conducted a numerical example to assess the efficiency of the proposed framework. Since we are dealing with uncertainty through the use of a sampling framework, we choose to generate a large number of instances (100 samples of 10 different sizes) by repeatedly sampling a first order autoregressive model. As our computational results suggest, our algorithm performed faster for this particular problem considered under a sampling framework. The experimental results show that, for a larger number of scenarios, the proposed algorithm can perform 4.5 times faster on average than solving the full-space equivalent deterministic problem. Moreover, our algorithm also presented better results when compared to other acceleration approaches considered.