An improved L-shaped method for two-stage convex 0-1 mixed integer nonlinear stochastic programs

Can Li[a], Ignacio E. Grossmann[a,*]

[a]*Center for Advanced Process Decision-Making, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, United States*

Abstract

In this paper, we propose an improved L-shaped method to solve large-scale two-stage convex 0-1 mixed-integer nonlinear stochastic programs with mixed-integer variables in both first and second stage decisions and with relatively complete recourse. To address the difficulties in solving large problems, we propose a Benders-like decomposition algorithm that includes both (strengthened) Benders cuts and Lagrangean cuts in the Benders master problem. The proposed algorithm is applied to solve a batch plant design problem under demand uncertainty and a planning problem under demand and price uncertainty. It is shown that the proposed algorithm outperforms the commercial solvers, DICOPT, SBB, Alpha-ECP, and BARON, for the problems with a large number of scenarios. Also it is proved that the proposed algorithm can yield a lower bound that is at least as tight as the one from Lagrangean decomposition algorithm.

Keywords: Stochastic programming; L-shaped method; Convex MINLP; Integer recourse

*Corresponding author.

E-mail address: grossmann@cmu.edu (I.E. Grossmann).

1 Introduction

Uncertainty is a crucial aspect in process systems engineering as there are often uncertainties present in parameters such as product demands and raw material prices (Grossmann et al., 2016). Optimizing for a given nominal or expected condition can lead to decisions that are suboptimal or deficient in robustness. Stochastic programming is an optimization framework that deals with decision making under uncertainty (Birge and Louveaux, 2011). In stochastic programming, it is assumed that the probability distributions of the uncertain parameters are known *a priori*. The uncertainties are typically approximated as some discrete realizations of the uncertain parameters as an approximation to the real continuous probability distribution. For example, the realizations of the demand of a product can have three different values which represent high, medium, and low demand, respectively. Each realization is defined as a scenario with a given probability. The objective of stochastic programming is to optimize the expected value of an objective function (e.g., the expected cost) over all the scenarios.

A special case of stochastic programming is two-stage stochastic programming. Specifically, stage 1 decisions are made 'here and now' at the beginning of the period, and are then followed by the resolution of uncertainty. Stage 2 decisions, 'wait and see' or recourse decisions, are taken as corrective action at the end of the period. One common type of two-stage stochastic program is 0-1 mixed-integer linear program with continuous recourse in the second stage presented in (1). Ω is the set of scenarios. τ_ω is the probability of scenario ω and $\sum_{\omega\in\Omega}\tau_\omega = 1$. x represent the first stage decisions. Let $I = \{1, \cdots, n\}$ be the index set of all the first stage variables. $I_1 \subseteq I$ is the subset for indices of the binary first stage variables. y_ω represents the second stage decisions in scenario ω. The uncertainties are reflected in the matrices (vectors), W_ω, h_ω, T_ω. (1) is often referred to as the deterministic equivalent or the extensive form of the two-stage stochastic program. Throughout this paper, we assume that the two-stage stochastic programs have relatively complete recourse, i.e., any solution x that satisfies the first stage constraints has feasible recourse decisions in the second stage. However, solving (1) directly can be prohibitive when the number of scenarios is large because the computational time generally grows exponentially with the number of scenarios.

$$\begin{aligned} \min \quad & z = c^T x + \sum_{\omega\in\Omega} \tau_\omega d_\omega^T y_\omega \\ & s.t. \quad Ax \le b \\ & W_\omega y_\omega \le h_\omega - T_\omega x \quad \forall \omega \in \Omega \\ & x \in \left\{x : x_i \in \{0,1\}, \forall i \in I_1,\ 0 \le x \le x^{ub}\right\} \\ & y_\omega \in \left\{y : y \ge 0\right\} \end{aligned} \tag{1}$$

Due to the difficulties in solving the deterministic equivalent problem, decomposition algorithms such as Lagrangean decomposition (Guignard, 2003; Oliveira et al., 2013), progressive hedging (Rockafellar and Wets, 1991), and Benders decomposition (Laporte and Louveaux, 1993; Van Slyke and Wets, 1969) can be applied to solve problem (1) more effectively. Lagrangean decomposition makes a copy of the first stage decisions for each scenario and adds non-anticipativity constraints ($NACs$) to ensure that the first stage decisions made for all the scenarios are the same. The $NACs$ are then dualized so that the deterministic equivalent problem is decomposed into scenarios which can be solved in parallel. The Lagrangean multipliers can be updated using the subgradient method or the cutting plane method (Oliveira et al., 2013; Kim and Zavala, 2015). A lower bound can be obtained at each iteration of Lagrangean decomposition, which is the summation of the optimal

value of the the Lagrangean subproblems. However, the upper bounding procedure of Lagrangean decomposition is in general a heuristic. For algorithmic advances and software implementation of the Lagrangean decomposition, we refer to the work of Kim and Zavala (2015).

Progressive hedging (Rockafellar and Wets, 1991) (PH) is a scenario decomposition method that is motivated by the augmented Lagrangean method (AL) and the alternating direction method of multipliers (ADMM) (Boyd et al., 2011). The $NACs$ are also dualized and quadratic terms are added to the objective function of each subproblem to penalize the deviation from the solution that satisfies the $NACs$. At each iteration of PH, a heuristic algorithm can be applied to obtain an upper bound. Gade et al. (2016) prove that a lower bound can be obtained from PH, which is similar to solving the Lagrangean subproblems. However, PH is only guaranteed to converge for problems that are continuous and convex (Rockafellar and Wets, 1991). Watson and Woodruff (2011) propose several heuristics to accelerate the convergence of PH for stochastic mixed-integer programs (SMIP). But PH is in general a heuristic algorithm for SMIP.

Benders decomposition (Laporte and Louveaux, 1993; Van Slyke and Wets, 1969), also referred to as L-shaped method in stochastic programming literature, starts by defining a master problem with only the first stage decisions. After the master problem is solved, the first stage decisions are fixed at the optimal solution of the master problem. Then the deterministic equivalent problem can be decomposed into $|\Omega|$ subproblems. The subproblems can be solved in parallel and valid inequalities of x can be derived and added to the Benders master problem. The master problem is solved again and the algorithm iterates until the upper bound and the lower bound converge. The lower bound is the optimal value of the Benders master problem. The upper bound is set to the best feasible solution. Benders decomposition is able to converge in a finite number of steps when the second stage decisions are all continuous and the second stage constraints are linear. An example is investment planning of process networks (Iyer and Grossmann, 1998) where the 0-1 variables are first stage decisions.

However, there are problems including 0-1 variables in the second stage decisions such as supply chain problems with fixed cost for transportation (Brunaud et al., 2017). For two-stage stochastic programs with integer or mix-integer recourse, Benders decomposition cannot be applied directly. Some work has been done to deal with two-stage mixed-integer linear stochastic programs with integer or mixed-integer recourse. In the pioneering work of Laporte and Louveaux (1993), a Benders-like algorithm with optimality cuts is proposed to solve a two-stage stochastic program with pure binary first stage variables.CarøE and Schultz (1999) propose a dual decomposition algorithm with branch and bound that is able to close the duality gap of Lagrangean relaxation. Others propose Benders-like decomposition algorithm with disjunctive cuts (Sen and Sherali, 2006; Ntaimo, 2010; Qi and Sen, 2017), or Gomory cuts (Gade et al., 2014). We refer to the survey of Küçükyavuz and Sen (2017) for recent advances.

For two-stage nonlinear mixed-integer stochastic programs, relatively little work has been done. Mijangos (2015) proposes an algorithm based on Branch-and-Fix Coordination method for convex problems with 0-1 mixed-integer variables in the first stage and only continuous variables in the second stage. Li et al. (2011) propose a nonconvex generalized Benders decomposition (NGBD) algorithm for mixed-integer nonlinear stochastic programs with pure binary first stage variables. Ogbe (2016) propose a joint decomposition method for mixed-integer nonlinear nonconvex programs. While the advantage of this algorithm is that it can globally optimize nonconvex two-stage stochastic programming problems, they do not handle mixed-integer recourse directly by fixing both the first stage variables and the nonconvex second stage variables when solving the Benders subproblems. The Benders master problem contains all the first stage variables and the nonconvex

second stage variables. Therefore, this algorithm may not scale well with the number of scenarios. Atakan and Sen (2017) propose a progressive hedging based branch-and-bound algorithm for two-stage convex 0-1 mixed integer stochastic programs with pure binary first stage variables.

To the best of our knowledge, there is no effective decomposition algorithm that can solve two-stage 0-1 mixed-integer convex nonlinear stochastic programs with mixed-integer variables in both first and second stage decisions. In this paper, we propose an improved L-shaped algorithm that has different types of valid inequalities in the Benders master problem. First of all, inspired by the work of Mitra et al. (2016) and Ogbe (2016), Lagrangean cuts are added to the Benders master problem to tighten the Benders master problem. Second, rank-one lift-and-project cuts are used to convexify the subproblems with mixed-integer variables and strengthened Benders cuts can be derived by solving the Benders subproblems with lift-and-project cuts. The paper is organized as follows: In section 2, we propose the improved L-shaped method. In section 3, two process systems engineering examples in this category are proposed as motivating examples. In section 4, computational experiments are performed to evaluate the performance of the algorithm.

2 The improved L-shaped algorithm

In problem (1), only linear constraints are included and the second stage variables are continuous. In this paper, we extend (1) by considering convex nonlinear constraints and mixed-integer variables in both first and second stage. The deterministic equivalent of the two-stage stochastic program we address in this paper is defined as (P) and shown in (2)-(7).

$$(P): \quad \min \quad z = c^T x + \sum_{\omega \in \Omega} \tau_\omega d_\omega^T y_\omega \tag{2}$$

$$s.t. \quad A_0 x \geq b_0, \quad g_0(x) \leq 0 \tag{3}$$

$$A_{1,\omega} x + g_{1,\omega}(y_\omega) \leq b_{1,\omega} \quad \forall \omega \in \Omega \tag{4}$$

$$g_{2,\omega}(y_\omega) \leq b_{2,\omega} \quad \forall \omega \in \Omega \tag{5}$$

$$x \in X, \quad X = \left\{x : x_i \in \{0,1\}, \forall i \in I_1, \ 0 \leq x \leq x^{ub}\right\} \tag{6}$$

$$y_\omega \in Y \quad \forall \omega \in \Omega, \quad Y = \left\{y : y_j \in \{0,1\}, \forall j \in J_1, \ 0 \leq y \leq y^{ub}\right\} \tag{7}$$

Here, x represents the first stage decisions. y_ω represents the second stage decisions in scenario ω. g_0, $g_{1,\omega}$, $g_{2,\omega}$, are convex functions. Both the first and the second stage decisions are mixed-integer. Let $I = \{1, \cdots, n\}$ be the index set of all the first stage variables. $I_1 \subseteq I$ is the subset for indices of the binary first stage variables. Let $J = \{1, \cdots, m\}$ be the index set of all the second stage variables. $J_1 \subseteq J$ is the subset for the indices of the binary second stage variables. x^{ub} is a vector that represents the upper bound of all the first stage variables. y^{ub} is a vector that represents the upper bound of all the second stage variables. In this paper, we assume problem (P) has relatively complete recourse. Solving (P) with a large number of scenarios directly using mixed-integer nonlinear programming (MINLP) solvers is prohibitive since the computational time grows exponentially with the number of scenarios. In order to solve problem (P) more efficiently, we propose an improved L-shaped method that is based on Benders decomposition (Van Slyke and Wets, 1969). As discussed in the introduction section, the basic idea of Benders decomposition is to decompose (P) into a Benders master problem which only contains first stage decisions and Benders subproblems each of which only contains second stage decisions for a given scenario. In

this paper, we add both (strengtened) Benders cuts and Lagrangean cuts to the Benders master problem to tighten the relaxation.

We first define the Benders master problem (MB^k), with both Lagrangean cuts (10), Benders cuts (11), and strengthened Benders cuts (12) for a given set of iterations $k \in K$. z^k_{MB} is the objective of the Benders master problem at iteration k. $z^{*k}_{SL,\omega}$ is the optimal value of the Lagrangean subproblem (SL^k_ω). $z^{*k}_{SB,\omega}$ and $z^{*k+}_{SSB,\omega}$ are the optimal values of the Benders subproblem (SB^k_ω) and strengthened Benders subproblem (SSB^{k+}_ω). All the subproblems are defined next. Note that (12) and (11) are the same type of cuts, i.e., (12) is a strengthened version of (11). Only one of (12) and (11) are included in the Benders master problem. $\tilde{x}^k$ represents the optimal solution to the Bender master problem in iteration k.

$$(MB^k): \quad \min \quad z^k_{MB} = \sum_\omega \eta_\omega \tag{8}$$

$$s.t. \quad A_0 x \geq b_0, \quad g_0(x) \leq 0 \tag{9}$$

$$\eta_\omega \geq z^{*k}_{SL,\omega} - \mu^k_\omega x \quad \forall \omega \in \Omega, k \in K \tag{10}$$

$$\eta_\omega \geq z^{*k}_{SB,\omega} + (\lambda^k_\omega)^T (x - \tilde{x}^k) + \tau_\omega c^T x \quad \forall \omega \in \Omega, k \in K \tag{11}$$

$$\eta_\omega \geq z^{*k+}_{SSB,\omega} + (\lambda^k_\omega)^T (x - \tilde{x}^k) + \tau_\omega c^T x \quad \forall \omega \in \Omega, k \in K \tag{12}$$

$$x \in X \tag{13}$$

Next, we show how the valid inequalities (10), (11) and (12) can be derived.

Lagrangean cuts. The deterministic equivalent problem (P) can be reformulated by duplicating the first stage decisions x for each scenario ω and adding nonanticipativity constraints ($NACs$), $x_{\omega 1} = x_{\omega 2}, x_{\omega 1} = x_{\omega 3}, \cdots, x_{\omega 1} = x_{\omega |\Omega|}$, to guarantee that all the first stage decisions made for all the scenarios are the same. The $NACs$ can be dualized by multiplying π_ω to the constraints, $x_{\omega 1} = x_{\omega+1}$, $\omega = \omega_1, \omega_2, \cdots, \omega_{|\Omega|-1}$. Then the deterministic equivalent problem can be decomposed into $|\Omega|$ scenarios. Each subproblem is defined as a Lagrangean subproblem (SL^k_ω) for iteration k:

$$(SL^k_\omega): \quad \min \quad z^k_{SL,\omega} = \tau_\omega (c^T x_\omega + d^T_\omega y_\omega) + \mu^k_\omega x_\omega \tag{14}$$

$$s.t. \quad A_0 x_\omega \geq b_0, \quad g_0(x_\omega) \leq 0 \tag{15}$$

$$A_{1,\omega} x_\omega + g_{1,\omega}(y_\omega) \leq b_{1,\omega} \tag{16}$$

$$g_{2,\omega}(y_\omega) \leq b_{2,\omega} \tag{17}$$

$$x_\omega \in X \tag{18}$$

$$y_\omega \in Y \tag{19}$$

where $\mu^k_{\omega 1} = \sum_{\omega=\omega 1}^{\omega_{|\Omega|-1}} \pi^k_\omega$, $\mu^k_{\omega+1} = -\pi^k_\omega$, $\omega = \omega_1, \omega_2, \cdots, \omega_{|\Omega|-1}$. Each Lagrangean subproblem (SL^k_ω) is solved to optimality. The optimal value of the subproblem ω at iteration k is defined as $z^{*k}_{SL,\omega}$. $\eta_\omega \geq z^{*k}_{SL,\omega} - \mu^k_\omega x$ is a valid inequality for the Benders master problem defined as a Lagrangean cut. The proof of the validity of the Lagrangean cuts is shown in proposition 1 in Appendix A. A similar proof can be found in Ogbe (2016). After solving the Lagrangean subproblems at each iteration k, the multipliers of the Lagrangean subproblems can be updated using the subgradient method (Oliveira et al., 2013), which is described in Appendix B. After adding the Lagrangean cuts to the

Benders master problem, the lower bound obtained by solving the Benders master problem is at least as tight as the lower bound that can be obtained by using Lagrangean decomposition. The proof is shown in proposition 2 in Appendix A.

Benders cuts. Assume the solution to the Benders master problem at iteration k is $\tilde{x}^k$, x is fixed at $\tilde{x}^k$ for the relaxation of the deterministic equivalent problem (RP). (RP) can be decomposed into $|\Omega|$ subproblems. Each subproblem is defined as the Benders subproblem (SB_ω^k):

$$(SB_\omega^k): \quad \min \quad z_{SB,\omega}^k = \tau_\omega d_\omega^T y_\omega \tag{20}$$

$$s.t. \quad x = \tilde{x}^k \tag{21}$$

$$A_{1,\omega}x + g_{1,\omega}(y_\omega) \leq 0 \tag{22}$$

$$g_{2,\omega}(y_\omega) \leq b_{2,\omega} \tag{23}$$

$$0 \leq y_\omega \leq y^{ub} \tag{24}$$

According to Geoffrion (1972), a Benders cut can be generated at iteration k:

$$\begin{aligned}\eta_\omega \geq\ & \tau_\omega d_\omega^T \tilde{y}_\omega^k + (\lambda_\omega^k)^T(x - \tilde{x}^k) + (\chi_\omega^k)^T(g_{1,\omega}(\tilde{y}_\omega^k) - b_{1,\omega} + A_{1,\omega}\tilde{x}^k) \\ & + (\nu_\omega^k)^T(g_{2,\omega}(\tilde{y}_\omega^k) - b_{2,\omega}) + \psi_{\omega l}^k(0 - \tilde{y}_\omega^k) + \psi_{\omega u}^k(\tilde{y}_\omega^k - y^{ub}) + \tau_\omega c^T x \\ & = z_{SB,\omega}^{*k} + (\lambda_\omega^k)^T(x - \tilde{x}^k) + \tau_\omega c^T x\end{aligned} \tag{25}$$

where $\lambda_\omega^k, \chi_\omega^k, \nu_\omega^k, \psi_{\omega l}^k, \psi_{\omega u}^k$, are the optimal dual multipliers for constraints (21)-(24) at iteration k. $z_{SB,\omega}^{*k}$ is the optimal value of (SB_ω^k). $\tilde{y}_\omega^k$ is the optimal solution of (SB_ω^k).

Strengthened Benders cuts. The strengthened Benders subproblem is defined as follows:

$$(SSB_\omega^{k-}): \quad \min \quad z_{SSB,\omega}^k = \tau_\omega d_\omega^T y_\omega \tag{26}$$

$$s.t. \quad x = \tilde{x}^k \tag{27}$$

$$A_{1,\omega}x + g_{1,\omega}(y_\omega) \leq 0 \tag{28}$$

$$g_{2,\omega}(y_\omega) \leq b_{2,\omega} \tag{29}$$

$$0 \leq y_\omega \leq y^{ub} \tag{30}$$

$$(\alpha_{1\omega}^{jk'})^T x + (\alpha_{2\omega}^{jk'})^T y_\omega \leq \beta_\omega^{jk'} \quad \forall j \in J_{1\omega}^{k'}, k' \leq k-1 \tag{31}$$

where (31) are the valid inequalities that have been derived before the kth iteration. In the first iteration, (SSB_ω^{k-}) reduces to SB_ω^k. The minus sign in the notation of (SSB_ω^{k-}) means that the valid inequalities for iteration k have not been derived. Later on, we will denote the strengthened Benders subproblem in iteration k that includes the valid inequalities of iteration k as (SSB_ω^{k+}). As (SSB_ω^{k-}) is a relaxation of the convex MINLP, some $(\tilde{y}_\omega^k)_j$, $j \in J_1$, may be fractional. However, the second stage variables $(y_\omega)_j$, $j \in J_1$ have to satisfy the binary constraints. Therefore, the following disjunctions hold for each $j \in J_1$, which means that the binary second stage variables can be either 0 or 1.

$$\begin{bmatrix} A_0x \geq b_0, \quad g_0(x) \leq 0 \\ A_{1,\omega}x + g_{1,\omega}(y_\omega) \leq b_{1,\omega} \\ g_{2,\omega}(y_\omega) \leq b_{2,\omega} \\ 0 \leq y_\omega \leq y^{ub} \\ (y_\omega)_j = 1 \end{bmatrix} \vee \begin{bmatrix} A_0x \geq b_0, \quad g_0(x) \leq 0 \\ A_{1,\omega}x + g_{1,\omega}(y_\omega) \leq b_{1,\omega} \\ g_{2,\omega}(y_\omega) \leq b_{2,\omega} \\ 0 \leq y_\omega \leq y^{ub} \\ (y_\omega)_j = 0 \end{bmatrix} \quad \forall j \in J_1 \tag{32}$$

From the disjunctions, valid inequalities called lift-and-project cuts can be derived, which were first proposed by Balas et al. (1993) in the MILP setting and extended to convex MINLP by Stubbs and Mehrotra (1999). In order to derive the lift-and-project cuts, in each iteration k for each scenario ω, the minimum distance problem (MD_ω^{jk}) is solved for all $j \in J_{1\omega}^k$ where $J_{1\omega}^k$ is the index set that includes the indices of all the fractional $(\tilde{y}_\omega^k)_j$, $j \in J_1$. The objective function is any norm of the difference between (x, y_ω) and the fractional solution $(\tilde{x}^k, \tilde{y}_\omega^k)$. (u^d, v_ω^d) is the disaggregated variable for disjunct d. γ_d, $d = 1, 2$, are variables between 0 and 1 that represent the weight of each disjunct. According to Ceria and Soares (1999), (34)-(41) is the convex hull representation of the disjunction that enforces that the binary variable $(y_\omega)_j$ should be either 0 or 1.

$$(MD_\omega^{jk}): \quad \min \quad ||(x, y_\omega) - (\tilde{x}^k, \tilde{y}_\omega^k)|| \tag{33}$$

$$s.t. \quad x = \sum_{d=1,2} u^d, \quad y_\omega = \sum_{d=1,2} v_\omega^d \tag{34}$$

$$\gamma_1 + \gamma_2 = 1, \quad \gamma_1, \gamma_2 \geq 0 \tag{35}$$

$$0 \leq v_\omega^d \leq y^{ub}\gamma_d, \quad 0 \leq u^d \leq x^{ub}\gamma_d \quad d = 1, 2 \tag{36}$$

$$A_0 u^d \geq b_0 \gamma_d, \quad \gamma_d g_0(u^d/\gamma_d) \leq 0 \quad d = 1, 2 \tag{37}$$

$$A_{1,\omega} u^d + \gamma_d g_{1,\omega}(v_\omega^d/\gamma_d) \leq b_{1,\omega}\gamma_d \quad d = 1, 2 \tag{38}$$

$$\gamma_d g_{2,\omega}(v_\omega^d/\gamma_d) \leq 0 \quad d = 1, 2 \tag{39}$$

$$(v_\omega^1)_j = 0 \tag{40}$$

$$(v_\omega^2)_j = \gamma_2 \tag{41}$$

Let $(\bar{x}_\omega^{jk*}, \bar{y}_\omega^{jk*})$ be the optimal solution of (MD_ω^{jk}). Stubbs and Mehrotra (1999) proved that $(\xi_\omega^{jk})^T \begin{pmatrix} x - \bar{x}_\omega^{jk*} \\ y_\omega - \bar{y}_\omega^{jk*} \end{pmatrix} \geq 0$ is a valid inequality for the fullspace problem (P) where ξ_ω^{jk} is one subgradient of the objective function evaluated at the optimal solution of (MD_ω^{jk}). The cut is able to cut off the fractional solution $(\tilde{x}^k, \tilde{y}_\omega^k)$, if the optimal distance is strictly greater than 0.

For 2-norm, $\xi_\omega^{jk} = 2 \begin{pmatrix} \bar{x}_\omega^{jk*} - \tilde{x}^k \\ \bar{y}_\omega^{jk*} - \tilde{y}_\omega^k \end{pmatrix}$. The appropriate subgradient for 1-norm and $\infty-$norm can be found in the work of Stubbs and Mehrotra (1999).

The nonlinear lift-and-project cuts can be expensive since they require to solve an NLP for each fractional binary variable for each scenario in each iteration k. Inspired by the work of Zhu and Kuno (2006) and Kılınç et al. (2017) who outer-approximate the convex nonlinear functions to avoid solving the NLPs, we outer-approximate the nonlinear constraints and solve the cut generating linear program $(CGLP_\omega^{jk})$ instead of (MD_ω^{jk}).

$$(CGLP_\omega^{jk}): \quad \min \quad ||(x, y_\omega) - (\tilde{x}^k, \tilde{y}_\omega^k)|| \tag{42}$$

$$s.t. \quad x = \sum_{d=1,2} u^d, \quad y_\omega = \sum_{d=1,2} v_\omega^d \tag{43}$$

$$\gamma_1 + \gamma_2 = 1 \tag{44}$$

$$0 \leq v_\omega^d \leq y^{ub}\gamma_d, \quad 0 \leq u^d \leq x^{ub}\gamma_d \quad d = 1, 2 \tag{45}$$

$$A_0 u^d \geq b_0 \gamma_d, \quad \gamma_d g_0(x^l) + \nabla g_0(x^l)^T \big(u^d - x^l \gamma_d\big) \leq 0 \quad d = 1, 2, l \in L \tag{46}$$

$$A_{1,\omega}u^d + \gamma_d g_{1,\omega}(y^l_\omega) + \nabla g_{1,\omega}(y^l_\omega)^T\big(v^d_\omega - y^l_\omega\gamma_d\big) \leq b_{1,\omega}\gamma_d \quad d = 1,2, l \in L \tag{47}$$

$$\gamma_d g_{2,\omega}(y^l_\omega) + \nabla g_{2,\omega}(y^l_\omega)^T\big(v^d_\omega - y^l_\omega\gamma_d\big) \leq 0 \quad d = 1,2, l \in L \tag{48}$$

$$(v^1_\omega)_j = 0 \tag{49}$$

$$(v^2_\omega)_j = \gamma_2 \tag{50}$$

where (x^l, y^l_ω) are the points that we choose to outer-approximate the nonlinear constraints. L is the index set of such points. Note that in order to solve an LP instead of an NLP for the cut generating process, we can only use 1-norm or ∞-norm in the objective of $(CGLP^{jk}_\omega)$. The lift-and-project cuts derived by solving $(CGLP^{jk}_\omega)$ are weaker than that derived by solving (MD^{jk}_ω), since the feasible region of $(CGLP^{jk}_\omega)$ is a relaxation of the feasible region of (MD^{jk}_ω). So there is a tradeoff between computational efficiency and the tightness of the cuts that are derived. For the problems that we investigate, the $(CGLP^{jk}_\omega)$ is able to provide good computational results, which will be discussed in section 4. For the derivation of the lift-and-project cuts based the solution of $(CGLP^{jk}_\omega)$, we refer to the work of Kılınç et al. (2017). Here, we simply denote the lift-and-project cut derived from $(CGLP^{jk}_\omega)$ as:

$$(\alpha^{jk}_{1\omega})^T x + (\alpha^{jk}_{2\omega})^T y_\omega \leq \beta^{jk}_\omega \tag{51}$$

The lift-and-project cuts that are obtained by solving (MD^{jk}_ω) or $(CGLP^{jk}_\omega)$ are added to the Benders subproblem (SB^k_ω). The strengthened Benders subproblem with the lift-and-project cuts that are derived in iteration k is defined as (SSB^{k+}_ω) where the plus sign simply means that the newly derived cuts are included:

$$(SSB^{k+}_\omega): \quad \min \quad z^k_{SSB,\omega} = \tau_\omega d^T_\omega y_\omega \tag{52}$$

$$s.t. \quad x = \tilde{x}^k \tag{53}$$

$$A_{1,\omega}x + g_{1,\omega}(y_\omega) \leq 0 \tag{54}$$

$$g_{2,\omega}(y_\omega) \leq b_{2,\omega} \tag{55}$$

$$0 \leq y_\omega \leq y^{ub} \tag{56}$$

$$(\alpha^{jk'}_{1\omega})^T x + (\alpha^{jk'}_{2\omega})^T y_\omega \leq \beta^{jk'}_\omega \quad \forall j \in J^{k'}_{1\omega}, k' \leq k \tag{57}$$

where (57) are the lift-and-project cuts that are derived until iteration k for scenario ω. A strengthened Benders cut can be derived by solving the strengthened Benders subproblem in the same way that we derive the Benders cuts:

$$\eta_\omega \geq z^{*k+}_{SSB,\omega} + (\lambda^k_\omega)^T(x - \tilde{x}^k) + \tau_\omega c^T x \tag{58}$$

Note that we slightly abuse notation here: λ^k_ω is used to represent the optimal dual variable for both the Benders subproblem (SB^k_ω) and the strengthened Benders subproblem (SSB^{k+}_ω).

Upper bounding procedure. After the Benders master problem is solved at iteration k, x in problem (P) is fixed at $\tilde{x}^k$. The rest of the problem can be decomposed into $|\Omega|$ scenarios. A feasible solution to the deterministic equivalent problem can be obtained by solving the upper bound subproblems (UB^k_ω) defined by (59)-(62). Let $z^{*k}_{UB,\omega}$ be the optimal value of (UB^k_ω). An upper bound of (P) can be obtained by calculating $c^T\tilde{x}^k + \sum_{\omega\in\Omega} z^{*k}_{UB,\omega}$.

$$(UB_\omega^k): \quad \min \quad z_{UB,\omega}^k = \tau_\omega d_\omega^T y_\omega \tag{59}$$

$$s.t. \quad g_{1,\omega}(y_\omega) \leq b_{1,\omega} - A_{1,\omega}\tilde{x}^k \tag{60}$$

$$g_{2,\omega}(y_\omega) \leq b_{2,\omega} \tag{61}$$

$$y_\omega \in Y \tag{62}$$

The steps of the improved L-shaped method are outlined in Figure 1. The lower bound is initialized to $-\infty$. The upper bound is initialized with the optimal value obtained by solving the stochastic program with the first stage decisions fixed at the optimal solution of the expected value problem (EEV) (Birge and Louveaux, 2011). The Lagrangean multipliers are initialized to 0. At each iteration, the Lagrangean subproblems, the Benders master problems, the strengthened Benders subproblems, the cut generating linear program, the strengthened Benders subproblem with the newly generated cuts, and the upper bound subproblems are solved sequentially. The optimal solution of the Benders master problem is used to initialize the strengthened Benders subproblems, the cut generating linear programs, the strengthened Benders subproblems with the newly generated cuts and the upper bound subproblems. Valid inequalities for the Benders master problem can be derived by solving the Lagrangean subproblems, the strengthened Benders subproblems with the newly generated cuts. The algorithm iterates until the relative optimality gap is within a given value ϵ.

However, for problems with good NLP relaxations, since the Benders cuts are already tight enough, we can just solve the Benders subproblems without generating the lift-and-project cuts. Note that in each iteration, we can add either Benders cuts or strengthened Benders cuts to the Benders master problem, but not both, because the strengthened Benders cuts are derived from strengthened Benders subproblems (SSB_ω^{k+}) which have tighter relaxations than the Benders subproblems SB_ω^k. Therefore, the strengthened Benders cuts dominate the Benders cuts.

The proposed algorithm

The steps of the improved L-shaped method are outlined below:

1. Initialization

Set $\mu_\omega^0 = 0$ for all $\omega \in \Omega$, $k = 0$, $LB = -\infty$, $UB = EEV$(expected result of using the expected value problem solution).

Go to Step 2.

2. Lagrangean subproblem

For fixed μ_ω^k, solve each Lagrangean subproblem SL_ω^k in parallel. The optimal solution is $\hat{x}_\omega^k$, $\hat{y}_\omega^k$, $z_{SL,\omega}^{*k}$.

Add the Lagrangean cuts, $\eta_\omega \geq z_{SL,\omega}^{*k} - \mu_\omega^k x$, to the Benders master problem.

If $\sum_{\omega\in\Omega} z_{SL,\omega}^{*k} \geq LB$, set $LB = \sum_{\omega\in\Omega} z_{SL,\omega}^{*k}$.

Update $\mu_\omega^k \to \mu_\omega^{k+1}$ using the subgradient method shown in Appendix B.

Go to Step 3.

3. Benders master problem

Solve the Benders master problem. Obtain optimal solution $\tilde{x}^k$ and z_{MB}^{*k}.

If $z_{MB}^{*k} \geq LB$, set $LB = z_{MB}^{*k}$.

Go to Step 4.

4. Strengthened Benders subproblem

Fix $x = \tilde{x}^k$. Solve each strengthened Benders subproblem SSB_ω^{k-} in parallel. Obtain optimal primal solution $\tilde{y}_\omega^k$, $z_{SSB,\omega}^{*k-}$ and optimal dual variable λ_ω^k.

If there is any fractional variable $(\tilde{y}_\omega^k)_j$, $j \in J_1$, denote the index set of fractional variables as $J_{1\omega}^k$

for all $\omega \in \Omega$ and go to Step 5.
Otherwise, add the strengthened Benders cuts, $\eta_\omega \geq z^{*k-}_{SSB,\omega} + (\lambda^k_\omega)^T(x - \tilde{x}^k) + \tau_\omega c^T x$, to the Benders master problem and go to Step 7.

5. Cut generating linear program

Solve the cut generating linear programs $(CGLP^{jk}_\omega)$ for all $\omega \in \Omega, j \in J^k_{1\omega}$, in parallel. Obtain the lift-and-project cuts, $(\alpha^{jk'}_{1\omega})^T x + (\alpha^{jk'}_{2\omega})^T y_\omega \leq \beta^{jk'}_\omega$, and add them to the strengthened Benders subproblem (SSB^{k-}_ω). The subproblem with the newly generated lift-and-project cuts is defined as (SSB^{k+}_ω).
Go to Step 6.

6. Strengthened Benders subproblem

Solve the strengthened Benders subproblem (SSB^{k+}_ω) in parallel. Obtain optimal primal objective value $z^{*k+}_{SSB,\omega}$ and optimal dual variable λ^k_ω.
Add the strengthened Benders cuts, $\eta_\omega \geq z^{*k+}_{SSB,\omega} + (\lambda^k_\omega)^T(x - \tilde{x}^k) + \tau_\omega c^T x$, to the Benders master problem and go to Step 7.

7. Upper bound subproblem

Fix $x = \tilde{x}^k$ in each upper bound subproblem and solve them in parallel. Obtain optimal solution to the upper bound subproblem $\bar{y}^k_\omega, z^{*k}_{UB,\omega}$.
If $c^T\tilde{x}^k + \sum_{\omega\in\Omega} z^{*k}_{UB,\omega} \leq UB$, set $UB = c^T\tilde{x}^k + \sum_{\omega\in\Omega} z^{*k}_{UB,\omega}$.

8. Optimality check

If $UB \leq (1+\epsilon)LB$, stop.
Else set $k = k+1$ and include in K; go back to Step 2.

Although the algorithm can always yield a lower bound that is at least as tight as using Lagrangean decomposition, it still does not have theoretical guarantee of convergence. An intuitive explanation would be that there is a duality gap if Lagrangean cuts alone are added to the Benders master problem. While the strengthened Benders cuts are tighter than the Benders cuts as rank-one lift-and-project cuts are added to convexify the MINLP subproblems, the strengthened Benders subproblems are rank-one lift-and-project closure but not the convex hull of the original convex MINLP subproblems. The rank-one lift-and-project cuts can close a significant portion of the integrality gap for the original convex MINLP subproblems but it does not have theoretical guarantee to close all the integrality gap. Therefore, while combining Lagrangean cuts and strengthened Benders cuts can make the relative optimality gap smaller than both the duality gap and the integrality gap, the proposed algorithm does not have guarantee to close the optimality gap.

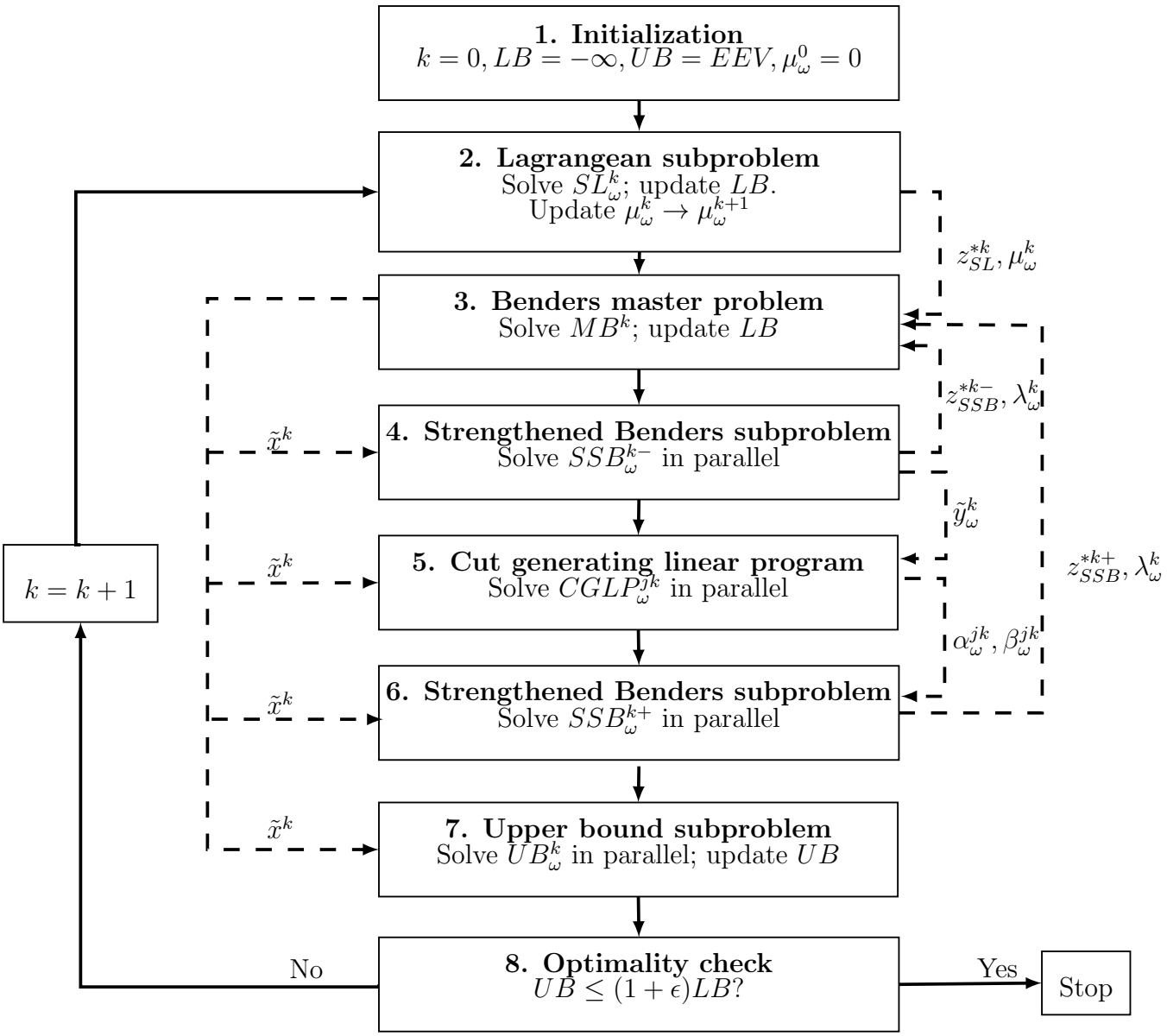

Figure 1. Algorithmic flow(solid line) and information flow(dotted line) of the improved L-shaped method

3 Motivating examples

3.1 Planning under demand uncertainty

Indices

i = process
j = chemical
s = production scheme
r = supplier
p = plant

c = customer
ω = scenario

Sets

$I(j)$ = index set of processes that consume chemical j
$O(j)$ = index set of processes that produce chemical j
$R(j)$ = index set of suppliers that provide chemical j
$PS(i)$ = index set of production schemes for process i
$JM(i,s)$ = index set of main products for production scheme s of process i
$L(i,s)$ = index set of chemicals involved in production scheme s of process i that are not main product for which the input-output relationship with the main product is linear
$\bar{L}(i,s)$ = index set of chemicals involved in production scheme s of process i that are not main product for which the input-output relationship with the main product is logarithmic

Parameters

Q_{pi}^{U} = maximum capacity of process i in plant p
μ_{ijs} = material balance coefficients for chemical j for production scheme s for process i
ρ_{ijs} = relative maximum production rate of main product j, $j \in JM(i,s)$, for production scheme s for continuous flexible process i referenced to a base scheme $\bar{s}$
H_i = maximum time for which process i is available for production
$D_{cj\omega}$ = demand of chemical j from customer c in scenario ω
α_i^C = fixed cost for installation of process i
β_i^C = variable cost for installation of process i
β_{rj}^S = price for purchase of chemical j from supplier r
β_{rp}^{RP} = variable cost for transporting chemical from supplier r to plant p
α_{rp}^{RP} = fixed cost for transporting chemical from supplier r to plant p
α_{pc}^{PC} = fixed cost for transporting chemical from plant p to customer c
β_{pc}^{PC} = variable cost for transporting chemical from plant p to customer c
δ_{is} = unit operating cost for production scheme s for process i
$\phi_{cj\omega}$ = penalty cost for not satisfying demand from customer c for chemical j in scenario ω
PU_{rpj}^{U} = maximum amount of chemical j transported between supplier r and plant j
F_{pcj}^{U} = maximum amount of chemical j transported between plant j and customer c
τ_ω = probability of scenario ω

First stage decisions

x_{pi} = binary variable (1 if process i is installed in plant p)
Q_{pi} = capacity of process i in plant p

Second stage decisions

$PU_{rpj\omega}$ = purchase amount of chemical j of plant p from supplier r in scenario ω
$y_{rp\omega}^{R}$ = binary variable (1 if any chemical is transported between supplier r and plant p in scenario ω)
$F_{pcj\omega}$ = amount of chemical j transported between plant p and customer c in scenario ω
$y_{pc\omega}^{C}$ = binary variable (1 if any chemical is transported between plant p and customer c in scenario ω)
$T_{pijs\omega}$ = amount of time assigned to the production of main product j for production scheme s of process i in plant p in scenario ω
$\theta_{pijs\omega}$ = amount of main product j, $j \in JM(i,s)$, produced from production scheme s of process i in plant p in scenario ω. The mathematical definition is shown in constraint (67)
$W_{pijs\omega}$ = amount of chemical j produced from production scheme s of process i in plant p in

scenario ω
$S_{cj\omega}$ = slack variable for not satisfying the demand of chemical j from customer c in scenario ω

3.1.1 Problem statement

We study a supply chain where raw materials are purchased from suppliers to manufacture products and deliver them to customers, see Figure 2. Each plant consists of a processing network with several processes interconnected in a finite number of ways. The processes can be dedicated or flexible (Norton and Grossmann, 1994). A flexible process can have multiple production schemes where these schemes can be different in raw materials, products or combination of both. A dedicated process has fixed recipe of raw materials and products, i.e., only one scheme. We assume all the processes considered in this paper are continuous. Therefore, process capacity can be expressed as production rate. There is a fixed and a variable cost associated with the installation of each process. The demands of customers are regarded as uncertain parameters, which will be realized after the investment decisions are made. If the manufacturer fails to satisfy the customer demands, extra products can be purchased from other manufacturers and there will be a penalty cost for the extra products purchased. The penalty cost can be regarded as the market prices of the products. In this problem, we assume that the prices of the products are another source of uncertainty, which will be realized after the first stage decisions are made. The second stage decisions include the purchase and transport of raw materials, the production amount in each scheme of each process in each plant, the delivery of products to customers and the purchase of products from other manufacturers. Note that there is a fixed cost for the transportation link between a supplier and a plant or a plant and a customer. Therefore, both the first stage and second stage variables are mixed-integer.

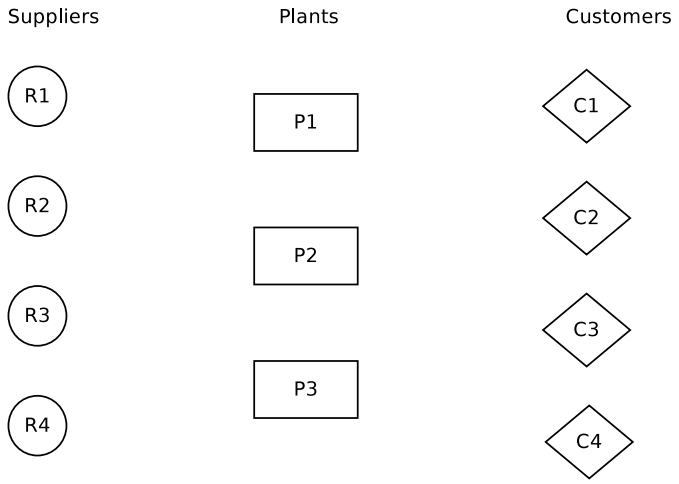

Figure 2. Suppliers, Plants, and Customers in Supply Chain

3.1.2 Mathematical formulation

The constraints for the investment decisions for each process i in each plant p are as follows:

$$Q_{pi} \leq Q_{pi}^{U} x_{pi} \quad \forall p, i \tag{63}$$

x_{pi} are binary variables to denote whether process i in plant p is installed. Constraint (63) will force the capacity to zero if x_{pi} is zero.

Constraint (64) represents the mass balance for each chemical j in a given plant p. The sum of purchases of raw materials from all suppliers plus the the sum of output of chemical j from all processes should be equal to the sum of delivery of j to all potential customers plus the sum of consumption of j from all processes.

$$\sum_{r \in R(j)} PU_{rpj\omega} + \sum_{i \in O(j)} \sum_{s \in PS(i)} W_{pijs\omega} = \sum_{c} F_{pcj\omega} + \sum_{i \in I(j)} \sum_{s \in PS(i)} W_{pijs\omega} \quad \forall p, j, \omega \tag{64}$$

The maximum production rate of a main product j for production scheme s is proportional to the capacity of the plant for a reference scheme $\bar{s}$. For $\bar{s}$, $\rho_{ij\bar{s}}, j \in JM(i, \bar{s})$ equals to 1. The amount produced can be calculated by the production rate times the production time.

$$W_{pijs\omega} = \rho_{ijs} Q_{pi} T_{pijs\omega} \quad \forall p, i, s \in PS(i), j \in JM(i, s), \omega \tag{65}$$

For each process i, the total production time has to be less than the available time of that process.

$$\sum_{s \in PS(i)} \sum_{j \in JM(i,s)} T_{pijs\omega} \leq H_i \quad \forall p, i, \omega \tag{66}$$

Note that constraint (65) is nonlinear since it includes the bilinear term $Q_{pi} T_{pijs\omega}$. To linearize it, we define new variable $\theta_{pijs\omega}$ in (67) and rewrite (65)-(66) as (68)-(69).

$$\theta_{pijs\omega} = Q_{pi} T_{pijs\omega} \quad \forall p, i, j \in JM(i, s), s \in PS(i), \omega \tag{67}$$

$$\sum_{s \in PS(i)} \sum_{j \in JM(i,s)} \theta_{pijs\omega} \leq H_i Q_{pi} \quad \forall p, i, \omega \tag{68}$$

$$W_{pijs\omega} = \rho_{ijs} \theta_{pijs\omega} \quad \forall p, i, s, s \in PS(i), j \in JM(i, s), \omega \tag{69}$$

Constraint (70) and (71) specifies the input-output relationship between the main product and other chemicals involved in scheme s of process i. Note that for some of the processing schemes, the input-output relationship is linear while others can be logarithmic. The physical meaning of the logarithmic relationship is that in some of the processes, the larger the input material flowrate, the less efficient those schemes become to produce products.

$$W_{pijs\omega} = \mu_{ijs} W_{pij's\omega} \quad \forall p, i, j \in L(i, s), j' \in JM(i, s), s \in PS(i), \omega \tag{70}$$

$$\ln(1 + W_{pijs\omega}) = \mu_{ijs} W_{pij's\omega} \quad \forall p, i, j \in \bar{L}(i, s), j' \in JM(i, s), s \in PS(i), \omega \tag{71}$$

Note that constraint (71) is a nonlinear equality which is nonconvex. This equality is relaxed to constraint (72) which is convex as pointed out by Kocis and Grossmann (1987). In the optimal

solution of the problem, constraint (72) is always active because the optimal solution will always tend to make as much product as possible.

$$\ln(1 + W_{pijs\omega}) \geq \mu_{ijs} W_{pij's\omega} \quad \forall p, i, j \in \bar{L}(i,s), j' \in JM(i,s), s \in PS(i), \omega \tag{72}$$

Binary variable $y^R_{rp\omega}$ denote whether a transportation link is established between supplier r and plant p, which will force the corresponding purchase amount to zero if the link is not established. Constraint (74) serves the same purpose for transportation between plants and customers.

$$PU_{rpj\omega} \leq PU^U_{rpj} y^R_{rp\omega} \quad \forall j, r, r \in R(j), p, \omega \tag{73}$$

$$F_{pcj\omega} \leq F^U_{pcj} y^C_{pc\omega} \quad \forall j, p, c, \omega \tag{74}$$

The demand of customer c for chemical j, $D_{cj\omega}$, must be satisfied by the total flow from all the plants to customer c plus the slack variable, $S_{cj\omega}$, which represents extra purchase of j from other companies if we fail to produce enough to satisfy the demand.

$$\sum_p F_{pcj\omega} + S_{cj\omega} = D_{cj\omega} \quad \forall c, j, \omega \tag{75}$$

The objective is to minimize the expected cost, which includes the investment cost, transportation cost, production cost, cost of purchasing raw materials, and penalty cost for not satisfying the demand.

$$\begin{aligned}
\min \quad Cost = & \sum_p \sum_i (\beta^C_i Q_{pi} + \alpha^C_i x_{pi}) + \sum_\omega \tau_\omega \Big(\sum_p \sum_i \sum_{s \in PS(i)} \delta_{is} \sum_{j \in JM(i,s)} \rho_{ijs} \theta_{pijs\omega} \\
& + \sum_p \sum_j \sum_{r \in R(j)} (\beta^S_{rj} + \beta^{RP}_{rp}) PU_{rpj\omega} + \sum_r \sum_p \alpha^{RP}_{rp} y^R_{rp\omega} \\
& + \sum_p \sum_c \alpha^{PC}_{pc} y^C_{pc\omega} + \sum_p \sum_c \sum_j \beta^{PC}_{pc} F_{pcj\omega} \\
& + \sum_c \sum_j \phi_{cj\omega} S_{cj\omega} \Big) \\
s.t. \quad & (63) - (64), (68) - (70), (72) - (75)
\end{aligned} \tag{76}$$

3.1.3 Case study

To illustrate the problem, we study a supply chain where we have 4 suppliers, 3 plants, 4 customers as in Figure 2. Each plant has the same process network superstructure as shown in Figure 3, which is from Norton and Grossmann (1994). As an example, process I1 is a dedicated continuous process producing chemical J3 from chemicals J1 and J6. As an example, process I2 is a flexible continuous process with two production schemes demonstrating product flexibility. Scheme S1 for process I2 produces chemical J3 from chemical J1, while scheme S2 produces chemical J4 from chemicals J1 and J6. The input-output relationship for process I1, I2, I3 is linear. The input-output relationship between chemical J3 and J5 for scheme S1 and between chemical J4 and chemical J5 for scheme S2 in process I4 is logarithmic. There are no processes initially installed.

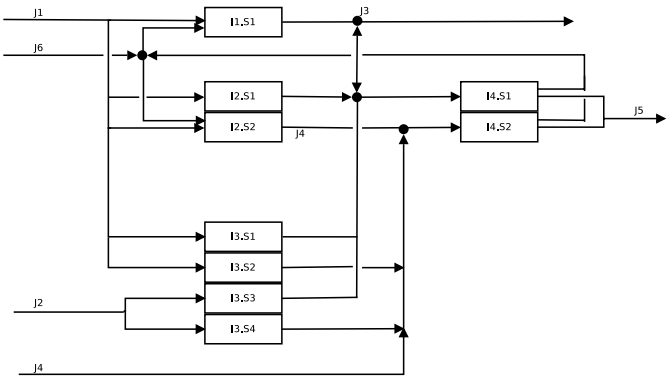

Figure 3. Process network for case study

We assume that there are 3 scenarios where the demand can be high, medium or low with probability of 25%, 50%, 25%, respectively. The high and low demand are +30% and -30% away from the medium. The convex MINLP problem is solved in deterministic equivalent form in GAMS using DICOPT on the 12 processors of an Intel Xeon (2.67GHz) machine with 64 GB RAM. The model has 84 binary variables, 2158 continuous variables, 2167 constraints. The optimal solution yields a minimum cost of 1509.3(10^6\$). The investment decisions for each of the three plants are shown in Table 1. The transportation links in three scenarios are shown in Figure 4. It is easy to see different recourse decisions are made in the three scenarios.

Table 1. Optimal capacity of each process in each plant

Q_{pi} (million lb/yr)	I1	I2	I3	I4
P1	150.0	0	0	4.9
P2	95.4	95.6	0	5.9
P3	57.5	36.1	0	5.5

To illustrate the capacity and material flows of process networks, the optimal configuration and mass flows of plant 1 in scenario 1 is shown in Figure 5. In this process network, processes I1 and I4 are installed. Raw material J1 is purchased from the suppliers and used as feedstock for I1.S1. Chemical J3 is produced by I1.S1. Part of J3 is delivered to customers directly while the rest is used as feedstock for I4.S1. Chemical J4 is purchased as raw material and used as feedstock

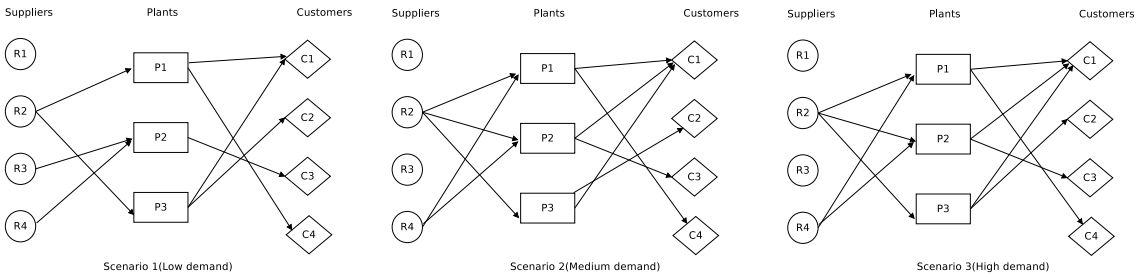

Figure 4. Optimal solution for transportation links

for I4.S2. Chemical J5 is produced by I4.S1 and I4.S2 and delivered to customers. Chemical J6 is produced by I4.S1 and I4.S2 and used as feedstock for I1.S1.

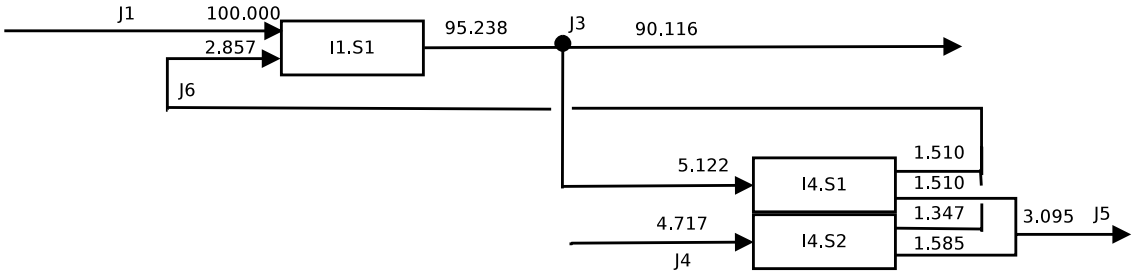

Figure 5. Optimal material flows for plant 1 in scenario 1

3.2 Optimal Design of Multi-product Batch Plant under Demand Uncertainty

Indices
i = products
j = stages
k = possible integer values for number of processing units
ω = scenarios
Parameters
$Q_{i\omega}$ = Demand of product i in scenario ω (kg)
S_{ij} = Size factor for product i in processing stage j (L/kg)
t_{ij} = Processing time for product i in processing stage j $(hours)$
H = Fixed production horizon time $(hours)$
α_i = Cost coefficient for purchasing unit in processing stage j ($)
β_j = Cost exponent for the volume of processing unit in stage j
λ_j = Fixed cost of using one unit in stage j
δ = Penalty cost for exceeding production horizon ($/hour)
p_ω = Probability of scenario ω
V^L = minimum volume of the processing units
V^U = maximum volume of the processing units
First stage decisions
V_j = Size of processing unit s for stage j
v_j = Nature logarithm of V_j

N_j^F = Number of processing units purchased for stage j
n_j^F = Natural logarithm of N_j^F
Y_{kj}^F = Binary variable. Equals 1 if $N_j^F = k$
Second stage decisions
$B_{i\omega}$ = Batch size of product i in scenario ω
$b_{i\omega}$ = Natural logarithm of $B_{i\omega}$
$TL_{i\omega}$ = Cycle time of product i in scenario ω
$tl_{i\omega}$ = Natural logarithm of $TL_{i\omega}$
$N_{j\omega}^S$ = Number of processing units that are operating in stage j in scenario ω
$n_{j\omega}^S$ = Natural logarithm of $N_{j\omega}^S$
$Y_{kj\omega}^S$ = Binary variable. Equals 1 if $N_{j\omega}^S = k$

3.2.1 Problem statement

We study a multi-product multi-stage batch plant (Figure 6) under demand uncertainty, which is an extension of the work of Grossmann and Sargent (1979). Every product has to be processed through all the stages. It is assumed that in each stage there can be multiple processing units with equal volume. The number and the volume of the processing units purchased have to be decided before the demand for each product is known. After the demand of each product is realized, operating decisions including the number of processing units actually used and the batch size of each product have to be made. We assume the batch sizes for any given product are the same. Not all the processing units purchased have to be used, i.e., processing units can be idle when the demand is low. There will be a fixed cost for using one processing unit, so one should try to use as few processing units as possible. One should also try to complete all the processing tasks within a given time horizon and there is a penalty cost for exceeding the time horizon to meet the demands. This problem can be formulated as a two-stage stochastic program where the first stage decisions are the number and volume of the processing units. The second stage decisions are the batch size of each product, the number of processing units used in each stage, and the cycle time of each product.

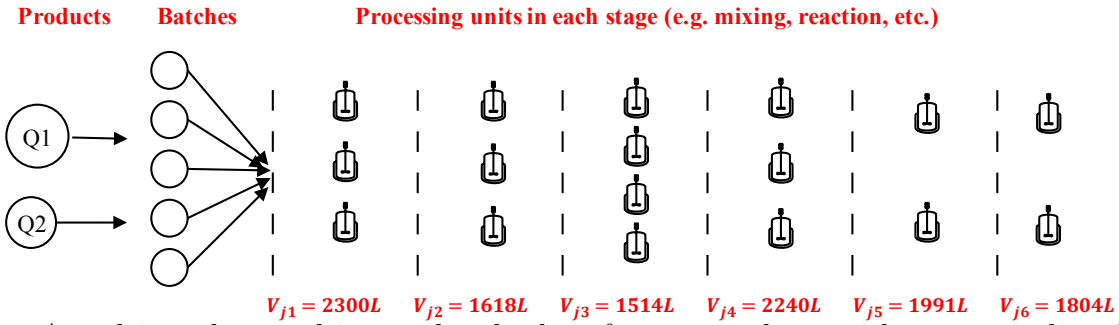

Figure 6. A multi-product multi-stage batch plant for two products with corresponding demands Q_1 and Q_2

3.2.2 Mathematical formulation

Constraints (77)-(78) determine the number of processing units purchased. Y_{kj}^F is the binary variable which equals to one when the number of processing units purchased at stage j, N_j^F, equals

to k.

$$\sum_k Y_{kj}^F = 1 \quad \forall j \tag{77}$$

$$N_j^F = \sum_k k Y_{kj}^F \quad \forall j \tag{78}$$

The volume of any processing unit has to be within V^L and V^U.

$$V^L \leq V_j \leq V^U \quad \forall j \tag{79}$$

Constraint (80) enforces the batch size in any scenario times a size factor cannot exceed the volume of the processing unit purchased in stage j.

$$V_j \geq S_{ij} B_{i\omega} \quad \forall i, j, \omega \tag{80}$$

$N_{j\omega}^S$ denotes the number of processing units that are operating in stage j in scenario ω. The binary variable $Y_{kj\omega}^S$ is used to guarantee the integrality of $N_{j\omega}^S$ (constraint (81) and (82)). The number of units that is operating in any scenario ω should be less than or equal to the number of units purchased, which is enforced by constraint (83).

$$\sum_k Y_{kj\omega}^S = 1 \quad \forall j, \omega \tag{81}$$

$$N_{j\omega}^S = \sum_k k Y_{kj\omega}^S \quad \forall j, \omega \tag{82}$$

$$N_{j\omega}^S \leq N_j^F \quad \forall j, \omega \tag{83}$$

The number of operating processing units times the cycle time should be greater or equal to any processing time t_{ij}.

$$N_{j\omega}^S T L_{i\omega} \geq t_{ij} \quad \forall i, j, \omega \tag{84}$$

The demand for product i in scenario ω is given by $Q_{i\omega}$ and the batch size is $B_{i\omega}$. Therefore, the number of batches is given by $\frac{Q_{i\omega}}{B_{i\omega}}$. The time consumed on producing product i equal the number of batches of product i times its cycle time. The summation over the processing time of all products should be less than or equal to the given time horizon plus slack variable L_ω representing the lateness in scenario ω.

$$\sum_i \frac{Q_{i\omega} T L_{i\omega}}{B_{i\omega}} \leq H + L_\omega \quad \forall \omega \tag{85}$$

The total cost includes the purchase of processing units, the fixed cost and variable cost of operating the processing units, and the penalty of lateness. As the variable cost of operating the processing units is always a constant which is proportional to the demand, it is not included in the objective.

$$cost = \sum_j \alpha_j N_j^F V_j^{\beta_j} + \sum_\omega p_\omega \Big(\sum_j \lambda_j N_{j\omega}^S + \delta L_\omega \Big) \tag{86}$$

The formulation, which includes constraints (77)-(86), is a nonconvex MINLP. However, according to Kocis and Grossmann (1988), this model can be reformulated as a convex MINLP

by considering exponential transformations of variables $V_j, N_j^F, B_{i\omega}, TL_{i\omega}$ and rewriting constraint (78)-(80), (82)-(86) as (87)-(94).

$$n_j^F = \sum_k \ln(k) Y_{kj}^F \quad \forall j \tag{87}$$

$$\ln(V^L) \leq v_j \leq \ln(V^U) \quad \forall j \tag{88}$$

$$v_j \geq \ln(S_{ij}) + b_{i\omega} \quad \forall i, j, \omega \tag{89}$$

$$n_{j\omega}^S = \sum_k \ln(k) Y_{kj\omega}^S \quad \forall j, \omega \tag{90}$$

$$n_{j\omega}^S \leq n_j^F \quad \forall j, \omega \tag{91}$$

$$n_{j\omega}^S + tl_{i\omega} \geq \ln(t_{ij}) \quad \forall i, j, \omega \tag{92}$$

$$\sum_i Q_{i,\omega} \exp(tl_{i\omega} - b_{i\omega}) \leq H + L_\omega \quad \forall \omega \tag{93}$$

This leads to the following convex MINLP:

$$\begin{aligned} \min \quad & cost = \sum_j \alpha_j \exp(n_j^F + \beta_j v_j) + \sum_\omega p_\omega \Big(\sum_j \lambda_j \exp(n_{j\omega}^S) + \delta L_\omega \Big) \\ s.t. \quad & (77), (81), (87) - (93) \end{aligned} \tag{94}$$

3.2.3 Case study

We study a batch plant with 6 stages, 5 products. Each stage can have at most 4 processing units. We assume that there are 3 scenarios, where the demand of each product can be high, medium, or low with probability of 25%, 50%, 25% respectively. The high and low demand are +10% and -10% away from the medium. The problem is first solved in deterministic equivalent form in GAMS using DICOPT on the 12 processors of an Intel Xeon (2.67GHz) machine with 64 GB RAM. The model has 96 binary variables, 64 continuous variables, 316 constraints. The minimum cost is 423.5(10^3\$). The NLP relaxation of this problem is 408.4(10^3\$). The number and volume of each processing units purchased in each stage is shown in Figure 6. The number of processing units that are operating in each stage in each scenario is shown in Table 2, where $\omega 1, \omega 2, \omega 3$ correspond to high, medium and low demand respectively. One can see that the recourse decisions of each scenario are different. When the demand is low, not all the processing units are used.

Table 2. Optimal number of processing units in each stage in each scenario

Stage	$j1$	$j2$	$j3$	$j4$	$j5$	$j6$
$N_{j\omega 1}^S$	3	3	4	3	2	2
$N_{j\omega 2}^S$	3	2	4	3	2	2
$N_{j\omega 3}^S$	3	2	4	2	2	2

However, if we solve the expected value problem with the demand of all products fixed at their mean values, the number of units purchased at each stage is 3, 2, 4, 3, 2, 2, respectively. Compared

to the stochastic solution, the number of units at stage $j2$ decreases from 3 to 2. Therefore, with the expected value solution, the high demand cannot be satisfied without an extension of the time horizon. The expected value solution has an expected cost of $433.4(10^3\$)$. The value of stochastic solution (VSS) (Birge and Louveaux, 2011) of this problem is $9.9(10^3\$)$.

4 Computational results

In order to test the proposed algorithm, the two proposed examples are solved with increasing number of scenarios. For the batch plant design problem under demand uncertainty, the scenarios are constructed by assuming the demands of all the products can be high, medium or low. We also assume that some of the product demands are independent of all the other product demands while some of the products demands vary simultaneously. For example, in the 81-scenario case, we assume that the demands of product $i1$, $i2$, and $i3$ are independent of the demands of all the other products while the demands of product $i4$ and $i5$ vary simultaneously. So there are $3^4 = 81$ scenarios. For the planning problem under demand and price uncertainty, we have two products $C3$ and $C5$. In the 3-scenario problem, we assume that the demands of product $C3$ and $C5$ vary simultaneously and they can be high, medium, or low. In the problems with 9, 27, and 81 scenarios, we assume that the demands and the prices of all the products can be high, medium, or low. The demands are always assumed to be independent of the prices. The prices of the two products $C3$ and $C5$ vary simultaneously in the problem with 9 and 27 scenarios but they are assumed to be independent of each other in the 81-scenario problem. The demands of $C3$ and $C5$ are assumed to vary simultaneously in the 9-scenario problem but they are assumed to be independent of each other in the problems with 27 and 81 scenarios. Note that in practice the discrete probability distribution can be constructed using historical data. If the computational resources are limited, the number of scenarios can be reduced using sample average approximation Kleywegt et al. (2002).

First of all, the deterministic equivalent of both the batch plant design problem and the planning problem with different number of scenarios are solved using different convex MINLP solvers including DICOPT, AlphaECP, SBB, and BARON on the 12 processors of an Intel Xeon (2.67GHz) machine with 64 GB RAM. The size of the deterministic equivalent problems, the wall time that the solvers require to solve the problems, the upper bound, the lower bound, the expected result of using the expected value problem solution (EEV), and value of stochastic solution(VSS) are shown in Table 3 and 4. It easy to see that the sizes of the problem can grow very quickly with the number of scenarios. For the problems with the largest number of scenarios, there can be thousands of binary variables and tens of thousands of constraints. Therefore, even the convex MINLP solver with the best performance, in the two cases, DICOPT, fails to solve the problems with a large number of scenarios within the 50,000 seconds time limit. The reason why DICOPT is so slow for the problems with a large number of scenarios is that the first master problem of the outer approximation algorithm is hard to solve to optimality. CPLEX cannot terminate the branch and bound within the time limit. The size of the node file is in the magnitude of tens of GB after compression at the time limit.

As the deterministic equivalent problems with a large number of scenarios are intractable, we implement different decomposition algorithms in GAMS. For the batch plant design problem, we have a tight NLP relaxation (see Table 3). The effects of the lift-and-project cuts are not significant

Table 3. Computational statistics of the deterministic equivalent of the batch plant design problem

Number of scenarios	3	27	81	243
Binary variables	96	672	1,968	5,856
Continuous variables	64	472	1,390	4,144
Constraints	316	2,158	6,424	19,222
Wall time(s) by BARON	2	210	50,000	$> 50,000$
Wall time(s) by SBB	4	99	9,602	$> 50,000$
Wall time(s) by Alpha-ECP	7	1,551	$> 50,000$	$> 50,000$
Wall time(s) by DICOPT	1	35	503	$> 50,000$
NLP relaxation (10^3\$)	408.4	394.5	391.7	391.0
Best upper bound (10^3\$)	423.5	415.9	413.6	-
Best lower bound (10^3\$)	423.5	415.9	413.6	-
Relative Optimality gap	0.0%	0.0%	0.0%	-
EEV (10^3\$)	433.4	421.8	418.8	418.2
VSS (10^3\$)	9.9	5.9	5.2	-

Table 4. Computational statistics of the deterministic equivalent of the planning problem

Number of scenarios	3	9	27	81
Binary variables	84	228	660	1,956
Continuous variables	2,158	6,424	19,222	57,616
Constraints	2,167	6,451	19,303	57,859
Wall time(s) by BARON	188	50,000	$50,000$	$> 50,000$
Wall time(s) by SBB	$> 50,000$	$> 50,000$	$> 50,000$	$> 50,000$
Wall time(s) by Alpha-ECP	462	$50,000$	$> 50,000$	$> 50,000$
Wall time(s) by DICOPT	24	$44,150$	$> 50,000$	$> 50,000$
NLP-relaxation (10^6\$)	1309.4	1309.2	1297.2	1297.2
Best upper bound (10^6\$)	1509.3	1508.8	-	-
Best lower bound (10^6\$)	1509.3	1508.8	-	-
Relative optimality gap	0.0%	0.0%	-	-
EEV (10^6\$)	1703.1	1702.6	1698.5	1698.5
VSS (10^6\$)	193.8	193.8	-	-

in terms of tightening the Benders subproblems. Therefore, for the batch plant design problem, we just solve the Benders subproblem without generating the lift-and-project cuts. Lagrangean decomposition and progressive hedging are also implemented. The multipliers of the Lagrangean decomposition algorithm are updated by the subgradient method shown in Appendix B. The ρ parameter in the progressive hedging algorithm is updated dynamically using the heuristic proposed by Watson and Woodruff (2011). In the progressive hedging algorithm, the lower bounds can be derived by neglecting the quadratic terms in the objective of the progressive hedging subproblems

and solving subproblems similar to the Lagrangean subproblems, which are proved to be valid lower bounds by Gade et al. (2016). The upper bounds of both the Lagrangean decomposition and the progressive hedging algorithm are derived by the nearest scenario heuristic, i.e., at each iteration, after the Lagrangean (progressive hedging) subproblems are solved to optimality, we take the weighted sum average of the first stage decisions, find the scenario whose first stage decisions are the closest to the weighted sum average, fix the first stage decisions at the first stage decisions of this scenario, and solve the upper bound subproblems in parallel to obtain a feasible solution.

As before, all the problems are solved using the 12 processors of an Intel Xeon (2.67GHz) machine with 64 GB RAM. All the MINLP problems are solved using DICOPT and all the NLP problems are solved using CONOPT. All the subproblems are solved in parallel using Grid GAMS (Bussieck et al., 2009). It is observed that the Lagrangean multipliers are close to convergence after the first 30 iterations. A similar trend hold for the progressive hedging algorithm. Therefore, we stop adding Lagrangean cuts after 30 iterations but keep adding Benders cuts until the lower bounds of the problem fail to improve for 10 iterations. The computational results of the decomposition algorithms for the batch problem are shown in Tables 5-8. L+B represents the algorithm with both Lagrangean cuts and Benders cuts in the Benders master problem. L represents the algorithm with only Lagrangean cuts in the Benders master problem. B represents the algorithm with only Benders cuts in the Benders master problem. LD represents the Lagrangean decomposition algorithm. PH represents the progressive hedging algorithm. The decomposition algorithms are all able to solve the problems within the time limit. For the batch problem with different number of scenarios, the proposed algorithm with both Lagrangean and Benders cuts can provide solutions with the smallest optimality gap. The lower bounds of the proposed algorithm are always tighter than the lower bounds given by LD and PH. In some cases, for example, the 3-scenario problem, LD or PH can give better feasible solutions than the proposed algorithm. However, in most cases (problems with 27, 81, and 243 scenarios), the best feasible solutions are given by the proposed algorithm. This comes at the expense of longer computational times. The CPU times of the subproblems and the master problems are also shown in Tables 5-8. Note that the CPU times are large for the problems with large number of scenarios. This is due to the mechanism of Grid GAMS that all the subproblems are submitted simultaneously, and thus each subproblem uses a small percentage of CPU. The CPU times are only able to show the relative expenses of the subproblems. For the Benders master problem, the CPU time is exactly what takes the 12 threads to solve the problem.

Table 5. Computational results of solving the 3-scenario batch plant design problem with different decomposition algorithms

3 scenario	L+B	L	B	LD	PH
Upper bound (10^3\$)	423.7	433.4	423.6	423.5	423.5
Lower bound (10^3\$)	423.3	422.7	421.6	419.5	408.4
Relative optimality gap	0.09%	2.51%	0.46%	0.96%	3.70%
Wall time (s)	53	27	26	13	42
Upper bound subproblem (s)	32	11	25	6	9
Lagrangean subproblem (s)	38	37	-	21	24
Benders subproblem (s)	1	-	1	-	-
Benders master problem (s)	10	3	5	-	-
Progressive hedging (s)	-	-	-	-	49

Table 6. Computational results of solving the 27-scenario batch plant design problem with different decomposition algorithms

27 scenario	L+B	L	B	LD	PH
Upper bound (10^3\$)	415.9	421.8	415.9	416.2	416.1
Lower bound (10^3\$)	413.7	404.7	413.6	403.6	404.2
Relative optimality gap	0.53%	4.21%	0.56%	3.12%	2.96%
Wall time (s)	228	66	110	37	56
Upper bound subproblem (s)	43	22	43	13	15
Lagrangean subproblem (s)	370	359	-	310	317
Benders subproblem (s)	7	-	7	-	-
Benders master problem (s)	134	15	52	-	-
Progressive hedging (s)	-	-	-	-	334

Table 7. Computational results of solving the 81-scenario batch plant design problem with different decomposition algorithms

81 scenario	L+B	L	B	LD	PH
Upper bound (10^3\$)	413.6	418.8	413.6	413.8	413.8
Lower bound (10^3\$)	411.4	402.9	411.3	402.6	403.2
Relative optimality gap	0.55%	3.96%	0.56%	2.78%	2.64%
Wall time (s)	574	146	352	86	129
Upper bound subproblem (s)	137	72	132	42	41
Lagrangean subproblem (s)	1015	1002	-	1018	913
Benders subproblem (s)	21	-	19	-	-
Benders master problem (s)	319	32	183	-	-
Progressive hedging (s)	-	-	-	-	967

The sizes of subproblems are shown in Table 9. The size of the Benders master problem corresponds to the size of the 243-scenario problem at the last iteration (59th iteration in this case).

Table 8. Computational results of solving the 243-scenario batch plant design problem with different decomposition algorithms

243 scenario	L+B	L	B	LD	PH
Upper bound (10^3\$)	413.1	418.2	413.1	413.1	413.3
Lower bound (10^3\$)	410.8	402.5	410.8	402.4	402.9
Relative optimality gap	0.55%	3.89%	0.56%	2.68%	2.58%
Wall time (s)	1962	490	609	270	376
Upper bound subproblem (s)	416	208	412	129	127
Lagrangean subproblem (s)	2557	2522	-	2657	2508
Benders subproblem (s)	64	-	56	-	-
Benders master problem (s)	1132	153	1310	-	-
Progressive hedging (s)	-	-	-	-	2621

Each subproblem is comparatively easier to solve than the deterministic equivalent problems.

Table 9. Size of each subproblem for the batch plant design problem

	Binary variables	Continuous variables	Constraints
Benders master problem	24	16	21,895
Upper bound subproblem	24	54	116
Benders subproblem	0	78	116
Lagrangean subproblem	48	30	104

For the planning problem, it is shown in Table 4 that the NLP relaxation is weak. Therefore, using the lift-and-project cuts to tighten the Benders subproblem should be effective. Different decomposition algorithms are implemented and the computational results are shown in Tables 10-13. SB+L represents the algorithm with both strengthened Benders cuts and Lagrangean cuts in the Benders master problem. SB represents the algorithm with only strengthened Benders cuts in the Benders master problem. Besides that, L, L+B, LD, and PH are also implemented as in the batch problem. All the decomposition algorithms are able to solve the planning problem within the time limit. Among all the algorithms, SB+L gives the smallest relative optimality gap in most cases. The algorithm with strengthened Benders cuts can give tighter optimality gap than the algorithm use Benders cuts. It can be observed that in most cases the strengthened Benders cuts can close the optimality gap of the algorithm that uses Benders cuts by more than one half. The algorithms that use strengthened Benders cuts or Lagrangean cuts can always yield a tighter lower bound than the Lagrangean decomposition or the progressive hedging algorithm. SB+L uses less time than SB in all the cases because adding Lagrangean cuts makes the problems take less iterations to converge. Although the computational time of SB+L is significantly higher than L+B, the reason is due to the fact that we have to solve a large number of cut generating linear programs in the SB+L case. It is shown in Tables 10-13 that the CPU time of the cut generating linear programs is small compared to other subproblems. Therefore, the significant increase in wall time is due to the data overhead of GAMS. If we can implement SB+L in languages like C++, we could expect the wall time to be much smaller. Note that in some cases, LD is able to give

good feasible solutions. However, the lower bound of LD is weaker than SB+L. In practice, if the decision-maker is not limited by computational resources, our proposed algorithm, SB+L, can provide the tightest lower bound and good feasible solutions. Otherwise, the decision-maker can also use Lagrangean decomposition to obtain good feasible solutions.

Table 10. Computational results of solving the 3-scenario planning problem with different decomposition algorithms

3 scenario	SB+L	L	SB	L+B	B	LD	PH
Upper bound (10^6\$)	1523.0	1703.1	1515.3	1537.4	1516.6	1532.8	1703.1
Lower bound (10^6\$)	1477.7	1472.1	1470.7	1472.4	1438.4	1471.9	1452.9
Relative optimality gap	3.07%	15.69%	3.03%	4.42%	5.44%	4.14%	17.22%
Wall time (s)	404	76	408	100	68	66	1384
Upper bound subproblem (s)	65	17	87	39	66	9	8
Lagrangean subproblem (s)	175	108	-	102	-	119	103
Benders subproblem (s)	-	-	-	6	7	-	-
Benders master problem (s)	35	6	30	15	18	-	-
Cut generating linear program (s)	112	-	156	-	-	-	-
Strengthened Benders subproblem (s)	54	-	73	-	-	-	-
Progressive Hedging (s)	-	-	-	-	-	-	1770

Table 11. Computational results of solving the 9-scenario planning problem with different decomposition algorithms

9 scenario	SB+L	L	SB	L+B	B	LD	PH
Upper bound (10^6\$)	1514.7	1702.6	1512.4	1528.6	1517.5	1545.5	1702.6
Lower bound (10^6\$)	1473.2	1439.3	1470.8	1456.5	1438.3	1438.0	1440.5
Relative optimality gap	2.82%	18.30%	2.83%	4.95%	5.51%	7.48%	18.19%
Wall time (s)	771	135	942	220	123	111	1695
Upper bound subproblem (s)	197	52	271	111	210	27	25
Lagrangean subproblem (s)	695	446	-	471	-	427	308
Benders subproblem (s)	-	-	-	20	37	-	-
Benders master problem (s)	62	7	96	57	51	-	-
Cut generating linear program (s)	394	-	506	-	-	-	-
Strengthened Benders subproblem (s)	174	-	243	-	-	-	-
Progressive Hedging (s)	-	-	-	-	-	-	4377

Table 12. Computational results of solving the 27-scenario planning problem with different decomposition algorithms

27 scenario	SB+L	L	SB	L+B	B	LD	PH
Upper bound (10^6\$)	1504.4	1698.5	1509.7	1516.0	1557.8	1509.0	1521.7
Lower bound (10^6\$)	1462.0	1422.2	1460.4	1441.7	1418.5	1420.7	1425.3
Relative optimality gap	2.90%	19.43%	3.38%	5.16%	9.82%	6.22%	6.76%
Wall time (s)	2115	234	2434	1448	156	209	2593
Upper bound subproblem (s)	577	5	757	27423	121	99	93
Lagrangean subproblem (s)	2114	2141	-	2123	-	2047	1384
Benders subproblem (s)	-	-	-	67	24	-	-
Benders master problem (s)	167	23	204	156	68	-	-
Cut generating linear program (s)	959	-	1311	-	-	-	-
Strengthened Benders subproblem (s)	404	-	584	-	-	-	-
Progressive Hedging (s)	-	-	-	-	-	-	18827

Table 13. Computational results of solving the 81-scenario planning problem with different decomposition algorithms

81 scenario	SB+L	L	SB	L+B	B	LD	PH
Upper bound (10^6\$)	1502.8	1698.5	1505.5	1512.7	1553.1	1509.3	1507.2
Lower bound (10^6\$)	1461.2	1418.5	1460.4	1441.5	1419.0	1416.7	1423.7
Relative optimality gap	2.85%	19.74%	3.09%	4.94%	9.45%	6.53%	5.87%
Wall time (s)	5986	555	10437	1239	729	348	4912
Upper bound subproblem (s)	1155	21	1690	1476	1589	260	362
Lagrangean subproblem (s)	8835	10100	-	10476	-	8164	5227
Benders subproblem (s)	-	-	-	222	138	-	-
Benders master problem (s)	372	80	604	570	443	-	-
Cut generating linear program (s)	2412	-	3506	-	-	-	-
Strengthened Benders subproblem (s)	975	-	2597	-	-	-	-
Progressive Hedging (s)	-	-	-	-	-	-	125690

The sizes of the subproblems and the master problem of the SB+L algorithm for the planning problem are shown in Table 14. The size of the Benders master problem and the strengthened Benders subproblem is the size of the 81-scenario problem at the last iteration (92th iteration in this case). Note that the strengthened Benders subproblem can have as many as 2,208 more constraints than the Benders subproblem at the last iteration. This corresponds to the rank-one lift-and-project cuts that are generated.

Table 14. Size of each subproblem for the planning problem

	Binary variables	Continuous variables	Constraints
Benders master problem	12	28	9,907
Upper bound subproblem	24	748	763
Benders subproblem	0	772	763
Strengthened Benders subproblem	0	772	2,971
Cut generating linear program	0	2,387	4,639
Lagrangean subproblem	36	736	739

5 Conclusion

In this paper, we have proposed an improved L-shaped method that can solve two-stage convex 0-1 mixed-integer nonlinear stochastic programs with mixed-integer variables in both first and second stage effectively. The proposed algorithm is a Benders-like decomposition algorithm that includes (strengthened) Benders cuts and Lagrangean cuts in the Benders master problem. The Benders cuts are derived by solving the Benders subproblems, which are the NLP relaxations of the subproblems for each scenario after the first stage decisions are fixed at the optimal solution of the Benders master problem. The strengthened Benders cut are derived by adding rank-one lift-and-project cuts to the Benders subproblem. The Lagrangean cuts are derived by solving the Lagrangean subproblems whose multipliers are updated by a new subgradient method.

Different decomposition algorithms including SB+L, L, SB, L+B, B, LD, PH are implemented and tested using the planning problem under demand and price uncertainty, and the batch plant design problem under demand uncertainty. It is shown that the combination of Lagrangean cuts and strengthened Benders cuts is able to provide the tightest lower bound among all the decomposition algorithms that we test although it requires more time than algorithms like LD. The proposed algorithm can be used as an effective computational strategy to solve the problems of the same category in PSE applications. However, more examples would be desirable to show the performance of the algorithm. Moreover, the proposed algorithm is not guaranteed to close the optimality gap of the problems due to the duality gap of the Lagrangean cuts and the integrality gap of the strengthened Benders cuts. A more rigorous algorithm that has finite convergence will be developed in the future.

6 Acknowledgment

The authors would like to acknowledge the support from CAPD at Carnegie Mellon University.

Appendix A

Proposition 1. *$\eta_\omega \geq z^{*k}_{SL,\omega} - \mu^k_\omega x$ is valid for the Benders master problem.*

Proof. We rewrite the full space problem (P) as (P') by introducing variables η_ω.

$$(P'): \quad \min \quad z = \sum_{\omega\in\Omega} \eta_\omega \tag{A.1}$$

$$s.t. \quad A_0 x \geq b_0, \quad g_0(x) \leq 0 \tag{A.2}$$

$$A_{1,\omega} x + g_{1,\omega}(y_\omega) \leq b_{1,\omega} \quad \forall \omega \in \Omega \tag{A.3}$$

$$g_{2,\omega}(y_\omega) \leq b_{2,\omega} \quad \forall \omega \in \Omega \tag{A.4}$$

$$\eta_\omega \geq \tau_\omega(c^T x + d^T_\omega y_\omega) \quad \forall \omega \in \Omega \tag{A.5}$$

$$x \in X \tag{A.6}$$

$$y_\omega \in Y \quad \forall \omega \in \Omega \tag{A.7}$$

It is easy to see (P) and (P') are equivalent. The idea of Benders decomposition is to project the feasible region that include $(x, \eta_\omega, y_\omega)$ onto the (x, η_ω) space. Therefore, valid constraints for Benders master problem can be derived from (A.2)-(A.7), which involve only variables x, η_ω. $\tau_\omega(c^T x + d^T_\omega y_\omega) + \mu^k x$ is the objective function of Lagrangean subproblem(SL^k_ω). This objective function is minimized over constraints (A.2)-(A.4),(A.6),(A.7) in (SL^k_ω). Thus, $z^{*k}_{SL,\omega} \leq \tau_\omega(c^T x + d^T_\omega y_\omega) + \mu^k x$, is a valid inequality if we have (A.2)-(A.4),(A.6),(A.7). By combining constraint (A.5), we can show that $\eta_\omega \geq z^{*k}_{SL,\omega} - \mu^k_\omega x$ is a valid inequality for the Benders master problem. □

Proposition 2. *The Benders master problem with the Lagrangean cuts (10) always yields a lower bound that is at least as tight as using Lagrangean decomposition.*

Proof. In Lagrangean decomposition, the best lower bound is given by solving problem (D)

$$\begin{aligned}(D): \quad \max_{\mu_\omega, \forall\omega\in\Omega} \quad \min_{x_\omega, y_\omega, \forall\omega\in\Omega} \quad & \sum_\omega \tau_\omega(c^T x_\omega + d^T_\omega y_\omega) + \mu_\omega x_\omega \\ & s.t. \quad (15) - (19) \quad \forall \omega \in \Omega\end{aligned} \tag{A.8}$$

Let μ^*_ω be the optimal dual multiplier to problem (D). Let x^*_ω, y^*_ω be the optimal primal solution when μ_ω is fixed at μ^*_ω. Let $z^{D*}_\omega = \tau_\omega(c^T x^*_\omega + d^T_\omega y^*_\omega) + \mu^*_\omega x^*_\omega$. In the proposed algorithm, the Lagrangean cuts $\eta_\omega \geq z^{D*}_\omega - \mu^*_\omega x$, $\forall \omega \in \Omega$ are added to the Benders master problem. $\sum_{\omega\in\Omega} \eta_\omega \geq \sum_{\omega\in\Omega} (z^{D*}_\omega - \mu^*_\omega x) = \sum_{\omega\in\Omega} z^{D*}_\omega$ is implied by the Lagrangean cuts. Therefore, the lower bound obtained from the Benders master problem is at least as tight as the lower bound obtained from Lagrangean decomposition. □

Appendix B

Here we describe the subgradient method (Oliveira et al., 2013) to update the Lagrangean multipliers. The nonanticipativity constraint is written as

$$x_{\omega 1} = x_{\omega 2}, x_{\omega 1} = x_{\omega 3}, \cdots, x_{\omega 1} = x_{\omega |\Omega|} \tag{B.1}$$

We define the multipliers associated with the NACs as $\pi_\omega, \omega = \omega_1, \omega_2 \cdots, \omega_{|\Omega|-1}$ The multipliers are updated by the formula:

$$\pi_\omega^{k+1} = \pi_\omega^k + t^k(\hat{x}_{\omega 1}^k - \hat{x}_{\omega+1}^k) \quad \omega = \omega_1, \omega_2, \cdots, \omega_{|\Omega|-1} \tag{B.2}$$

$\hat{x}_\omega^k$ is the optimal solution to the Lagrangean subproblem SL_ω^k. The value of t^k is calculated by the formula:

$$t_k = \frac{\alpha_k(UB - LB)}{\sum_{\omega=\omega_1}^{\omega_{|\Omega|-1}} ||\hat{x}_{\omega 1}^k - \hat{x}_{\omega+1}^k||_2^2} \tag{B.3}$$

α_k is a scalar chosen between 0 and 2. The value of α_k is divided by a constant greater than 1 when $\sum_\omega z_{SL,\omega}^{*k}$ fails to improve.

References

Atakan, S. and Sen, S. (2017). A progressive hedging based branch-and-bound algorithm for stochastic mixed-integer programs. *Optimization Online*.

Balas, E., Ceria, S., and Cornuéjols, G. (1993). A lift-and-project cutting plane algorithm for mixed 0–1 programs. *Mathematical Programming*, 58(1-3):295–324.

Birge, J. R. and Louveaux, F. (2011). *Introduction to stochastic programming*. Springer Science & Business Media.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122.

Brunaud, B., Bassett, M. H., Agarwal, A., Wassick, J. M., and Grossmann, I. E. (2017). Efficient formulations for dynamic warehouse location under discrete transportation costs. *Computers & Chemical Engineering*, in press.

Bussieck, M. R., Ferris, M. C., and Meeraus, A. (2009). Grid-enabled optimization with GAMS. *INFORMS Journal on Computing*, 21(3):349–362.

CarøE, C. C. and Schultz, R. (1999). Dual decomposition in stochastic integer programming. *Operations Research Letters*, 24(1):37–45.

Ceria, S. and Soares, J. (1999). Convex programming for disjunctive convex optimization. *Mathematical Programming*, 86(3):595–614.

Gade, D., Hackebeil, G., Ryan, S. M., Watson, J.-P., Wets, R. J.-B., and Woodruff, D. L. (2016). Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs. *Mathematical Programming*, 157(1):47–67.

Gade, D., Küçükyavuz, S., and Sen, S. (2014). Decomposition algorithms with parametric gomory cuts for two-stage stochastic integer programs. *Mathematical Programming*, 144(1-2):39–64.

Geoffrion, A. M. (1972). Generalized benders decomposition. *Journal of Optimization Theory and Applications*, 10(4):237–260.

Grossmann, I. E., Apap, R. M., Calfa, B. A., Garcia-Herreros, P., and Zhang, Q. (2016). Recent advances in mathematical programming techniques for the optimization of process systems under uncertainty. *Computers & Chemical Engineering*, 91:3–14.

Grossmann, I. E. and Sargent, R. W. (1979). Optimum design of multipurpose chemical plants. *Industrial & Engineering Chemistry Process Design and Development*, 18(2):343–348.

Guignard, M. (2003). Lagrangean relaxation. *Top*, 11(2):151–200.

Iyer, R. R. and Grossmann, I. E. (1998). A bilevel decomposition algorithm for long-range planning of process networks. *Industrial & Engineering Chemistry Research*, 37(2):474–481.

Kılınç, M. R., Linderoth, J., and Luedtke, J. (2017). Lift-and-project cuts for convex mixed integer nonlinear programs. *Mathematical Programming Computation*, pages 1–28.

Kim, K. and Zavala, V. M. (2015). Algorithmic innovations and software for the dual decomposition method applied to stochastic mixed-integer programs. *Optimization Online*.

Kleywegt, A. J., Shapiro, A., and Homem-de Mello, T. (2002). The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502.

Kocis, G. R. and Grossmann, I. E. (1987). Relaxation strategy for the structural optimization of process flow sheets. *Industrial & Engineering Chemistry Research*, 26(9):1869–1880.

Kocis, G. R. and Grossmann, I. E. (1988). Global optimization of nonconvex mixed-integer nonlinear programming (minlp) problems in process synthesis. *Industrial & Engineering Chemistry Research*, 27(8):1407–1421.

Küçükyavuz, S. and Sen, S. (2017). An introduction to two-stage stochastic mixed-integer programming. In *Leading Developments from INFORMS Communities*, pages 1–27. INFORMS.

Laporte, G. and Louveaux, F. V. (1993). The integer l-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133–142.

Li, X., Tomasgard, A., and Barton, P. I. (2011). Nonconvex generalized benders decomposition for stochastic separable mixed-integer nonlinear programs. *Journal of Optimization Theory and Applications*, 151(3):425.

Mijangos, E. (2015). An algorithm for two-stage stochastic mixed-integer nonlinear convex problems. *Annals of Operations Research*, 235(1):581–598.

Mitra, S., Garcia-Herreros, P., and Grossmann, I. E. (2016). A cross-decomposition scheme with integrated primal–dual multi-cuts for two-stage stochastic programming investment planning problems. *Mathematical Programming*, 157(1):95–119.

Norton, L. C. and Grossmann, I. E. (1994). Strategic planning model for complete process flexibility. *Industrial & Engineering Chemistry Research*, 33(1):69–76.

Ntaimo, L. (2010). Disjunctive decomposition for two-stage stochastic mixed-binary programs with random recourse. *Operations Research*, 58(1):229–243.

Ogbe, E. (2016). *New rigorous decomposition methods for mixed-integer linear and nonlinear programming*. PhD thesis, Queen's University (Canada).

Oliveira, F., Gupta, V., Hamacher, S., and Grossmann, I. E. (2013). A lagrangean decomposition approach for oil supply chain investment planning under uncertainty with risk considerations. *Computers & Chemical Engineering*, 50:184–195.

Qi, Y. and Sen, S. (2017). The ancestral benders' cutting plane algorithm with multi-term disjunctions for mixed-integer recourse decisions in stochastic programming. *Mathematical Programming*, 161(1-2):193–235.

Rockafellar, R. T. and Wets, R. J.-B. (1991). Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147.

Sen, S. and Sherali, H. D. (2006). Decomposition with branch-and-cut approaches for two-stage stochastic mixed-integer programming. *Mathematical Programming*, 106(2):203–223.

Stubbs, R. A. and Mehrotra, S. (1999). A branch-and-cut method for 0-1 mixed convex programming. *Mathematical Programming*, 86(3):515–532.

Van Slyke, R. M. and Wets, R. (1969). L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663.

Watson, J.-P. and Woodruff, D. L. (2011). Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science*, 8(4):355–370.

Zhu, Y. and Kuno, T. (2006). A disjunctive cutting-plane-based branch-and-cut algorithm for 0- 1 mixed-integer convex nonlinear programs. *Industrial & Engineering Chemistry Research*, 45(1):187–196.