A Branch-and-Bound Method for Discretely-Constrained Mathematical Programs with Equilibrium Constraints*

Yohan Shim, Marte Fodstad, Steven A. Gabriel, Asgeir Tomasgard

July 23, 2017

Abstract

We present a branch-and-bound algorithm for discretely-constrained mathematical programs with equilibrium constraints (DC-MPEC). This is a class of bilevel programs with an integer program in the upper-level and a complementarity problem in the lower-level. The algorithm builds on the work by Gabriel et al. (2010) and uses Benders decomposition to form a master problem and a subproblem. The new dynamic partition scheme that we present ensures that the algorithm converges to the global optimum. Partitioning is done to overcome the non-convexity of the Benders subproblem. In addition Lagrangean relaxation provides bounds that enable fathoming in the branching tree and warm-starting the Benders algorithm. Numerical tests show significantly reduced solution times compared to the original algorithm. When the lower level problem is stochastic our algorithm can easily be further decomposed using scenario decomposition. This is demonstrated on a realistic case.

1 Introduction

In this paper, we focus on bilevel programming problems where the upper-level deals with discrete decisions and the lower-level is a mixed complementarity problem (MCP). It is a variant of the traditional mathematical

*The paper is published in Annals of Operations Research, Nov 2013, **210** (1), pp 5-31. The final publication is available at Springer via http://dx.doi.org/10.1007/s10479-012-1191-5

program with equilibrium constraints (MPEC) where the leader is only allowed to make discrete decisions. We call the whole formulation DC-MPEC (Gabriel et al. 2010).

Gabriel et al. (2010) propose a heuristic to solve the DC-MPEC problem based on Benders decomposition. They rephrase the problem as a mixed integer linear problem (MILP) and decompose the problem by placing all constraints and objective elements containing lower-level variables in the Benders sub problem. The master problem domain is a priori heuristically partitioned into subdomains of x with the aim of finding subdomains where the lower level objective is convex. Afterwards each subdomain is solved by Benders decomposition method. It is shown that the heuristic can give a sub-optimal solution unless all subdomains are convex.

In our new approach we develop an idea mentioned in Gabriel et al. (2010) with a dynamic branching procedure that partitions the subdomains as the algorithm proceeds. We branch until the subdomains which have candidates for the global solution are convex and thereby guarantee to find the optimal solution. As opposed to the branching on single variables as employed in many branch-and-bound approaches, we use intersection of Benders cuts to partition the upper-level decision domain. The branching procedure is supported by lower bounding based on Lagrangean relaxation, which makes it possible to cut off parts of the master problem domain and thereby increase the efficiency of the algorithm compared to the static version. Using LR to accelerate the branch-and-bound procedure was introduced in both Geoffrion (1974) for MILPs and Falk (1969) for non-convex programs.

Several papers contribute to improving the convergence properties of Benders decomposition, for a nice review see Saharidis & Ierapetritou (2010). One common approach is to add cuts to the relaxed master problem, as for instance Saharidis et al. (2011) do. Similarly we utilize the solution value from the Lagrangean relaxation as a bound in the Benders decomposition somewhat inspired by cross decomposition (van Roy 1983, 1986). To the best of our knowledge this is a new way of utilizing Lagrangean relaxation results in bilevel programming: using it both in the lower bounding in the branch-and- bound procedure and to accelerate the Benders decomposition used to find the upper bound.

We also show how the addition of strong duality constraints, enabled by the Benders decomposition, increases the robustness of the transformation from DC-MPEC to MILP. When the lower-level is a two-stage stochastic MCP, we show how the lower bounding method can be adapted using scenario decomposition (Carøe & Schultz 1999) to achieve further decomposition, and test this on a natural gas application.

Our computational results show that using the dynamic partitioning al-

gorithm supported by the strong duality constraints considerably reduces the partitioning work needed compared to the static version of the algorithm.

Even the continuous linear bilevel programming problem has been shown to be NP-hard (Hansen et al. 1992) and the discrete nature of upper-level variables and their related constraints would make the DC-MPEC problem even more intractable. A substantial number of contributions exist for the different problem classes within bilevel programming, and we will point out the ones closest related to our DC-MPEC problem. For a broader overview see for instance Dempe (2002) or Colson et al. (2007) on bilevel programming and Luo et al. (1996), Outrata et al. (1998) or Fukushima & Lin (2004) on MPEC.

Gabriel & Leuthold (2010) formulate a Stackelberg game within the electric power market as a DC-MPEC and provide exact solutions with standard branch-and-bound after reformulating to a MILP. On the contrary most solution procedures for DC-MPEC are heuristics. Meng et al. (2009) and Meng & Wang (2011) use genetic algorithms supplemented with suitable procedures for solving lower level parametric VIs for facility location and service network design applications, respectively. Another DC-MPEC application is presented by Wang & Lo (2008) who transforms their problem into a mixed integer nonlinear program solved with an application specific heuristic.

Mesbah et al. (2011) use generalized Benders decomposition to solve a bilevel problem for transportation network design. Their upper level has binary variables and a non-linear objective function. The lower level consists of three parts, two optimization problems and an equilibrium problem. Lagrangean relaxation is used to solve the optimization lower-level problems.

Saharidis & Ierapetritou (2009) propose Benders decomposition for problems closely related to our DC-MPEC, but with lower level limited to LPs. They decompose the problem into a master problem containing all integer variables and pure integer constraints and a bilevel subproblem. The subproblem is transformed into a single level problem by using the KKT conditions and provides a feasibility cut or optimality cut for the master problem in each iteration. The integrality conditions are handled by adding integer exclusion cuts to the master problem. This work differs from ours in different ways of decomposing the problem and different ways of treating the integrality requirements.

Wen & Yang (1990) also solve a DC-MPEC with the lower level limited to LPs. They do not use the common reformulation to MILP based on KKT conditions, but develop valid bounds adapted to the bilevel structure and apply these in branch-and-bound. A similar strategy is used by Moore & Bard (1990) for bilevel problem with integrality constraints in both upper and lower level, and they also point out why standard bounding and fathoming

rules for branch-and-bound in integer programming do not apply for their problem.

The rest of this paper is organized as follows: First we present the basic ideas of the algorithm, with lower bounding, upper bounding and dynamic partitioning. Then follows a pseudocode overview of the total algorithm and proofs for the validity of bounds and overall convergence. The section ends with a description on how to adapt the lower bounding method to stochastic programs. Next follows numerical results on general problems with randomly generated data and on a natural gas supply chain problem before we conclude.

2 Dynamic Algorithm

The overall discretely-constrained mathematical program with equilibrium constraints (DC-MPEC) is given as follows:

$$\begin{aligned} &\min_{x,y} c^\top x + d^\top y \\ &s.t.\ Qx \leq q \\ &\quad Ax + By \geq a \\ &\quad y \in S(x) \end{aligned} \tag{1}$$

where $x \in Z^{n_x}$ and $y \in R^{n_y}$ are integer upper-level variables and continuous lower-level variables, respectively. The constraints $Qx \leq q$ contain the bounds on the x's and other linear constraints with only x variables; $Ax + By \geq a$ are the joint linear constraints upon x and y . The solution set of the lower-level MCP is given by

$$S(x) = \left\{ (y,z,w) \left| \begin{array}{r} 0 \leq y \perp Ey + e - M^\top z - D^\top w \geq 0 \\ 0 \leq z \perp My + Nx - k \geq 0 \\ Dy + Fx = g \end{array} \right. \right\} \tag{2}$$

where $z \in R^{n_z}$, $z \geq 0$ and $w \in R^{n_w}$. z and w are lower-level variables that typically correspond to dual variables of a underlying optimization problem. It is assumed that the dimension of each data element (i.e. $c, d, Q, q, A, B, a, E, e, M, N, k, D, F, g$) agrees with its associated variables. E is a symmetric and positive semi-definite matrix of the convex quadratic function $\frac{1}{2}y^\top Ey + e^\top y$ so that the KKT conditions are necessary and sufficient optimality conditions. Note that the lower-level problem also covers linear problems (LP) and quadratic convex problems (QP) since the KKT optimality conditions of these problem classes are MCP problems. We assume that a solution to Problem (1) exists.

As previously shown by amongst others Fortuny-Amat & McCarl (1981) a MPEC can be rephrased to a MILP through replacing the lower-level complementarity conditions by disjunctive constraints, binary variables and a large constant C. This reformulation implies an optimistic view on the lower-level in the sense that if multiple equilibria exist in lower-level the most favorable according to the upper-level objective is chosen. We denote the problem resulting from this reformulation (MILP).

We decompose (MILP) with Benders decomposition into a master problem:

$$
\begin{aligned}
&(\mathbf{B-MILP}) \\
&\min_x z_{\text{B-MILP}} = c^\top x + \alpha(x) \\
&s.t.\ Qx \leq q \\
&\breve{Q}x \leq \breve{q}
\end{aligned}
\tag{3}
$$

and a subproblem:

$$
\begin{aligned}
\alpha(x) = \min_{y,z,w,\bar{b},\tilde{b}}\ & d^\top y \\
s.t.\ & a \leq Ax + By \\
& 0 \leq y \leq C(1-\bar{b}) \\
& 0 \leq Ey + e - M^\top z - D^\top w \leq C\bar{b} \\
& 0 \leq z \leq C(1-\tilde{b}) \\
& 0 \leq My + (Nx - k) \leq C\tilde{b} \\
& Dy + Fx = g \\
& \bar{b}, \tilde{b} : \text{binary} \\
& y, z \geq 0
\end{aligned}
\tag{4}
$$

Only x variables are placed in the master problem and the other variables are in the subproblem. Because of the disjunctive variables $\bar{b}$ and $\tilde{b}$ the function $\alpha(x)$ is piecewise linear but not in general convex in x (Gabriel et al. 2010). The non-convexity means Benders decomposition method is not guaranteed to converge to an optimal solution for (MILP) (Benders 1962). The main idea of the dynamic partitioning algorithm is to partition the domain of $X = \{x \in Z^{n_x} | Qx \leq q\}$ into subdomains where $\alpha(x)$ is convex, as illustrated in Figure 1. The partitioning is controlled by upper and lower bounds that will converge as the non-convexities are removed in exchange for an increasing number of subdomains. $\breve{Q}x \leq \breve{q}$ is a set of linear partitioning constraints defining a subdomain. An overview of the problems used in this paper and their relations is given in Figure 2.

2.1 Lower Bounding

Traditionally the LP relaxation is used for bounding in branch-and-bound, but the MILP reformulation with binary variables and large constants gives weak LP bounds. Instead we apply the Lagrangean relaxation algorithm (LR) as lower bound of (MILP) relaxing $Qx \leq q$ with μ as the Lagrangean

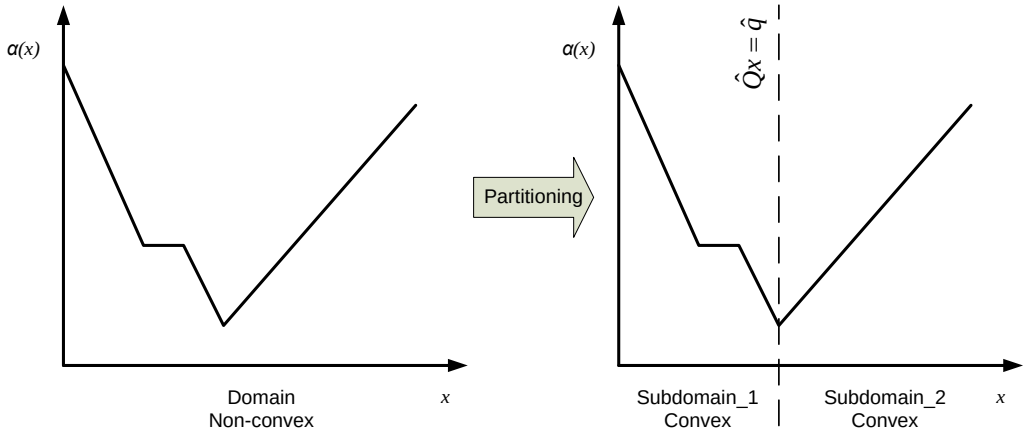

Figure 1: Illustration of how partitioning transforms a domain with a non-convex function into two subdomains with convex functions

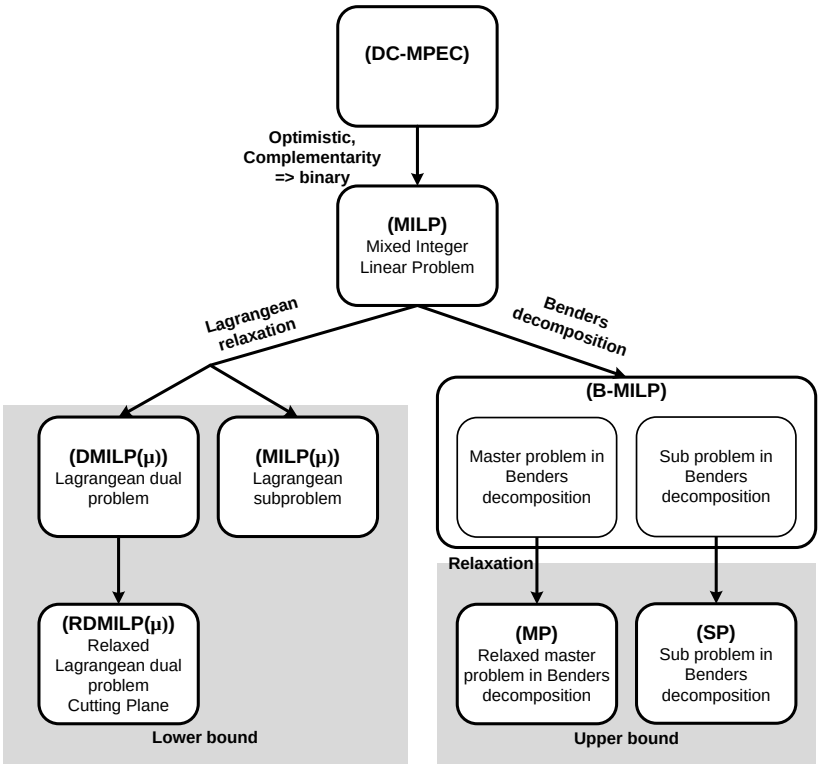

Figure 2: Problem and subproblem overview

multiplier. The mathematical formulation of the Lagrangean subproblem (MILP(μ)) is given as Problem (5) and its dual problem is defined as $z_{\text{LD}} = \max_{\mu}\{\phi(\mu)|\mu \geq 0\}$. (MILP($\mu$)) is a relaxation of (MILP) as proved in Geoffrion (1974).

$$
\begin{aligned}
&(\mathbf{MILP}(\mu)) \\
\phi(\mu) = \min_{x,y,z,w,\bar{b},\tilde{b}} & z_{\text{MILP}(\mu)} = c^\top x + d^\top y + \mu^\top (Qx - q) \\
s.t.\ & \breve{Q}x \leq \breve{q} \\
& a \leq Ax + By \\
& 0 \leq y \leq C(1 - \bar{b}) \\
& 0 \leq Ey + e - M^\top z - D^\top w \leq C\bar{b} \\
& 0 \leq z \leq C(1 - \tilde{b}) \\
& 0 \leq My + (Nx - k) \leq C\tilde{b} \\
& Dy + Fx = g \\
& x : \text{integer} \\
& \bar{b}, \tilde{b} : \text{binary} \\
& y, z \geq 0
\end{aligned}
\tag{5}
$$

MILP(μ)) is a mixed integer linear program that usually will be solved repeatedly to make the Lagrangean process converge. This means (MILP(μ)) needs to be significantly simpler than (MILP) to solve to make the lower bounding worthwhile. Generally that means there should be a substantial number of constraints $Qx \leq q$ or these constraints should have a complicating structure (see Conejo et al. 2006) as in the example of stochastic programming in Section 3.2.

In our implementation of LR, the Lagrangean multipliers (μ) are updated by a cutting plane method (Conejo et al. 2006). The Lagrangean iterations are stopped whenever the gap between the cutting plane problem (relaxed Lagrangean dual problem) and the Lagrangean subproblem (MILP(μ)) are sufficiently small. Also a limit on the number of iterations is implemented to avoid any cycling. A duality gap can occur because the Lagrangean subproblem has integral variables, as shown in Geoffrion (1974). This represents a non-convexity domain for $\alpha(x)$ that will cause further partitioning.

2.2 Upper Bounding

We apply Benders decomposition method (BD) as described in Conejo et al. (2006) to measure the upper bound (UB) of (MILP). The function $\alpha(x)$ in the master problem (3) is relaxed, and through the solution procedure rebuilt by iteratively solving the relaxed master problem (MP) and sub problem (SP) below.

$$
\begin{aligned}
&(\mathbf{MP}) \\
&\min_{x} c^{\top}x + \alpha \\
&s.t.\ Qx \leq q \\
&\quad \breve{Q}x \leq \breve{q} \\
&\quad \alpha \geq \alpha(x^{(k)}) + \lambda^{\top}\left(x - x^{(k)}\right) \\
&\quad \alpha \geq \alpha^{\text{down}} \\
&\quad k = 1, \ldots, v-1
\end{aligned}
\tag{6}
$$

$$
\begin{aligned}
&(\mathbf{SP}) \\
&\alpha(x) = \min_{y,z,w,\bar{b},\tilde{b}} d^{\top}y \\
&s.t.\ a \leq Ax + By \\
&\quad 0 \leq y \leq C(1-\bar{b}) \\
&\quad 0 \leq Ey + e - M^{\top}z - D^{\top}w \leq C\bar{b} \\
&\quad 0 \leq z \leq C(1-\tilde{b}) \\
&\quad 0 \leq My + (Nx - k) \leq C\tilde{b} \\
&\quad Dy + Fx = g \\
&\quad \bar{b}, \tilde{b} : \text{binary} \\
&\quad y, z \geq 0 \\
&\quad x^{(v)} \quad : (\lambda : \text{free})
\end{aligned}
\tag{7}
$$

In each Benders iteration, k, the solution of (SP) for a given $x^{(k)}$ gives a new Benders cut $\alpha \geq \alpha\left(x^{(k)}\right) + \lambda^{\top}\left(x - x^{(k)}\right)$ that is added to (MP) to approximate $\alpha(x)$. Let $z^{\text{down}}\left(x^{(v)}\right) = c^{\top}x^{(v)} + \alpha\left(x^{(v)}\right)$ and $z^{\text{up}}\left(x^{(v)}\right) = c^{\top}x^{(v)} + d^{\top}y^{(v)}$. Figure 3 illustrates a non-convex $\alpha(x)$ function and a master problem approximation based on a single Benders cut (broken line).

- If $\alpha(x)$ is convex for a given subdomain, Benders cuts create a lower envelope of $\alpha(x)$. In each iteration v, (MP) and (SP) provide lower

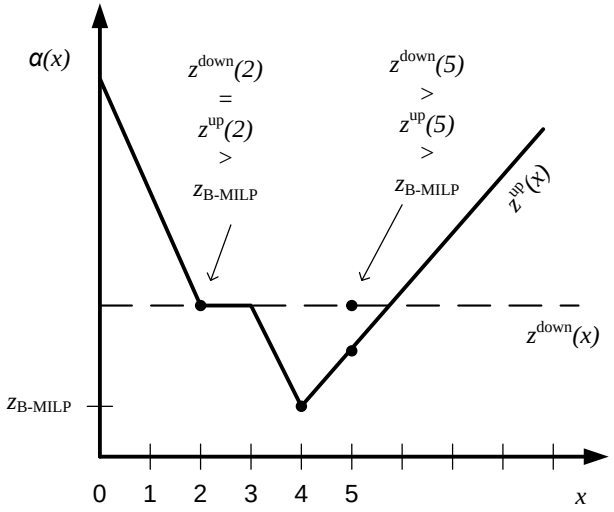

Figure 3: Illustration of non-convex $\alpha(x)$ and a single Benders cut (broken line)

and upper bound on $z_{\mathrm{B-MILP}}$, respectively, $z^{\mathrm{down}}\left(x^{(v)}\right) \leq z_{\mathrm{B-MILP}} \leq z^{\mathrm{up}}\left(x^{(v)}\right)$ and these bounds iteratively converge (Conejo et al. 2006).

- If $\alpha(x)$ is non-convex for a given subdomain, the Benders cuts may overestimate $\alpha(x)$ and eliminate a true optimum in the subdomain. This implies that either $z^{\mathrm{down}}\left(x^{(v)}\right)$ and $z^{\mathrm{up}}\left(x^{(v)}\right)$ converge to a value greater than the true optimum or $z^{\mathrm{down}}\left(x^{(v)}\right) > z^{\mathrm{up}}\left(x^{(v)}\right)$ which stops the iterations. These situations are illustrated in Figure 3 for $x = 2$ and $x = 5$, respectively.

In either case $z^{\mathrm{up}}\left(x^{(v)}\right)$ gives a valid upper bound, which is justified by the proof in the Appendix. Because of assumption (A2) that follows on page 18, we do not consider feasibility cuts.

2.2.1 Accelerating the Benders Decomposition by Including the Lower Bound

We could utilize the solution from (MILP(μ)) to warm-start the Benders iterations seeking to reduce the number of Benders iterations. The solution of x from (MILP(μ)) are set as the inital solution of (MP) and the objective value of (MILP(μ)) forms a lower bound for (MP), expressed by the optimality cut $c^{\top}x + \alpha \geq LB$. The optimality cut removes solutions inferior to the incumbent in the current iteration of the Benders decomposition, thereby reducing the search space and potentially producing faster convergence of the

Table 1: Numerical results: Speeding up UB measure by warm-staring with LB solution. The left part gives the dimensions of (MP) and (SP), where $dim(b) = dim(\bar{b}) + dim(\tilde{b})$. C is the value of the disjunctive constant. The right part gives the number of iterations and solution time in seconds for computing UB without and with warm-start (ws).

MP	SP			C	UB without ws		UB with ws	
x	y	z	b		Iter	Time	Iter	Time
20	100	100	200	1E5	14	18	6	7
20	100	100	200	1E5	14	17	6	7
20	100	100	200	1E5	14	18	5	6
20	100	100	200	1E5	10	12	4	5
20	100	100	200	1E5	10	12	4	5
20	1 000	1 000	2 000	1E6	130	938	74	470
20	1 000	1 000	2 000	1E6	77	508	46	307
20	1 000	1 000	2 000	1E6	14	87	3	18
20	1 000	1 000	2 000	1E6	23	144	3	19
20	1 000	1 000	2 000	1E6	118	854	37	251

algorithm. Table 1 compares the number of iterations and computation time of BD for two cases: without and with the warm-start (ws). Ten test problems were solved. All data were generated from the intervals $[0, 100]$ with uniform distributions. The upper-level decision variables were all binary; the lower-level problems are built by deriving the KKT conditions of LP problems. Matlab (ver. 7.0) and Xpress-MP (ver. 2006) were used to implement both BD and LR algorithms. LB and UB converged to the same value for all instances. As can be seen from the last two columns, the warm-start greatly reduced the number of iterations as well as the computation time.

2.2.2 Strong Duality Constraint

The complementarity constraints in the lower-level problem were linearized with disjunctive binary variables and big constant C in Gabriel et al. (2010), Gabriel & Leuthold (2010), Labbé et al. (1998), Hu et al. (2008), Mitsos (2010), Saharidis & Ierapetritou (2009) and DeNegre & Ralphs (2009), where their common question was the value of C for which the feasible region formed by complementarity constraints is not altered. Hu et al. (2008) proposed a solution method which does not require knowing the big constants, but the method is limited to linear programs with linear complementarity constraints (LPCC). Gabriel et al. (2010) shows that C can be chosen by a sensitivity

analysis or when the matrix M has a specific property the constants can be chosen analytically.

For lower-level problems with $E = 0$ (defined in Problem (2)), which for instance correspond to the KKT conditions of an LP, we impose the strong duality constraint $\left(e^\top - w^\top D\right) y = z^\top (k - Nx)$ to (B-MILP). The constraint was induced from the two complementarity constraints $y^\top \left(e - M^\top z - D^\top w\right) = 0$ and $z^\top (My + Nx - k) = 0$. A similar strong duality constraint cannot be imposed when $E \neq 0$ since that would give a non-linear constraint. This constraint is also not applicable if the lower-level problem does not contain the pair of complementarity constraints, which may be the situation if the problem has equilibria that are not derived from an underlying optimization problem.

Tables 2 and 3 contain results from testing the use of the strong duality constraint. We compare the valid range of C in (B-MILP) with strong duality constraint and (MILP) without this constraint. The valid range is the range where the optimal objective value of the original bilevel problem is reproduced. GAMS (ver. 23.0) and Xpress-MP (ver. 2006) were used to compute the solutions by enumeration. The bar graphs in the tables represent the effective range of C for each test problem. The strong duality constraint played a major role in making (B-MILP) almost insensitive to the choice of C.

The single-level approach in Gabriel & Leuthold (2010) and Audet et al. (2007), which corresponds to (MILP) may not use the strong duality constraint since it would make the problem non-linear and non-convex because x variables in the constraint cannot be fixed. Hence the working range of C for the single level approach would be narrow compared to the decomposed problem.

2.3 Branch-and-Bound

Each node in the branch-and-bound tree represents a subdomain for the upper-level variables x. The literature shows several ways to do branching of the subdomains, for instance in Bard & Moore (1990), Hansen et al. (1992) and Gabriel et al. (2010). We have chosen to follow Gabriel et al. (2010). Two sample points s_i (for $i = 1, 2$) are picked and Benders cuts, $\alpha(x) \geq \alpha(s_i) + \lambda_i^\top (x - s_i)$, are calculated for each point. λ_i is the dual variable vector to the linear constraint $x = s_i$ of (SP). Two planes are obtained by changing the inequality symbols of the Benders cuts to equality, that is, $\alpha(x) = \alpha(s_i) + \lambda_i^\top (x - s_i)$, and where these planes intersect, $\alpha(s_1) + \lambda_1^\top (x - s_1) = \alpha(s_2) + \lambda_2^\top (x - s_2)$, the domain is partitioned. Since $\alpha(x)$ can be non-convex, the two Benders cuts might not partition the domain into two

Table 2: The working range for C increases when strong duality constaints are added. For each instance the objective value, shape of the $\alpha(x)$-function and working range within [1E4,1E10] for (B-MILP) with strong duality constraints and (MILP) is given. The integer upper-level variable is limited to $[-10, 10]$, all model parameters are in $[0, 10]$ and $\dim(x) = 1$, $\dim(y) = 10$, $\dim(z) = 5$, $\dim(\bar{b}) + \dim(\tilde{b}) = 15$

ObjVal	Formulation	$\alpha(x)$	C (1E+n for n given below)							
			3	4	5	6	7	8	9	10
-37.00	B-MILP	non-convex								
	MILP	-								
-0.33	B-MILP	non-convex								
	MILP	-								
-33.75	B-MILP	non-convex								
	MILP	-								
3.00	B-MILP	non-convex								
	MILP	-								
-54.17	B-MILP	non-convex								
	MILP	-								
2.52	B-MILP	non-convex								
	MILP	-								
-1.33	B-MILP	non-convex								
	MILP	-								
-0.67	B-MILP	non-convex								
	MILP	-								
-6.53	B-MILP	non-convex								
	MILP	-								
-5.58	B-MILP	non-convex								
	MILP	-								
-0.88	B-MILP	non-convex								
	MILP	-								
-7.76	B-MILP	non-convex								
	MILP	-								
-19.52	B-MILP	non-convex								
	MILP	-								
0.00	B-MILP	non-convex								
	MILP	-								
-10.96	B-MILP	non-convex								
	MILP	-								

Table 3: The working range for C increases when strong duality constaints are added. For each instance the objective value, shape of the $\alpha(x)$-function and working range within [1E4,1E10] for (B-MILP) with strong duality constraints and (MILP) is given. The integer upper-level variables are limited to $[-10, 10]$, all model parameters are in $[0, 100]$ and $\dim(x) = 2$, $(y) = 400$, $\dim(z) = 200$, $\dim(\bar{b}) + \dim(\tilde{b}) = 600$

ObjVal	Formulation	$\alpha(x)$	C (1E+n for n given below) 4 5 6 7 8 9 10
302.72	B-MILP	non-convex	
	MILP	-	
517.00	B-MILP	convex	
	MILP	-	
2440.00	B-MILP	non-convex	
	MILP	-	
454.54	B-MILP	convex	
	MILP	-	
882.32	B-MILP	non-convex	
	MILP	-	
172.21	B-MILP	convex	
	MILP	-	
216.35	B-MILP	non-convex	
	MILP	-	
181.90	B-MILP	convex	
	MILP	-	
145.35	B-MILP	convex	
	MILP	-	
1423.00	B-MILP	non-convex	
	MILP	-	

non-empty subdomains. In that case new sample points are chosen a limited number of times, and if proper branching is still not achieved an arbitrary partition of the subdomain into two non-empty subdomains is chosen.

We use the following branching and fathoming rules:

- Branch if $|(UB - LB)/LB| > TOL$ and $LB < Incumbent$
- Pruning by optimality if $|(UB - LB)/LB| \leq TOL$
- Pruning by bound if $LB \geq Incumbent$

Here $Incumbent$ is the value of the best solution found so far and TOL is a user-specific tolerance.

2.4 Pseudocode for the Algorithm

Algorithm 1 is a pseudocode that describes the overall algorithm. The main workload is the branch-and-bound process, where N refers to the index of current node, L is the list of active nodes in the partitioning tree and $D(N)$ refers to the subdomain defined by node N. Three subroutines are used for solving the subproblems, making bounding decisions and branching and these are described in Subroutines 1-3. `SelectNextNode(`L`)` is a subroutine that selects next node from L. In our tests `SelectNextNode(`L`)` has applied the depth-first principle.

Algorithm 1 `Main`

1: $L := \{0\}$
2: $D(0) := X$
3: $Incumbent := \infty$
4: **while** $L \neq \emptyset$ **do**
5: $N :=$ `SelectNextNode(`L`)`
6: $L := L \setminus \{N\}$
7: $(Decision, Incumbent) :=$ `SolveAndBound(`$N, D(N), Incumbent$`)`
8: **if** $Decision =$ BRANCHING **then**
9: $(N{+}1, N{+}2, D(N{+}1), D(N{+}2)) :=$ `DecomposeDomain(`$N, D(N)$`)`
10: $L := L \cup \{N{+}1, N{+}2\}$
11: **end if**
12: **end while**
13: **return** $(Incumbent, (x^*, y^*, z^*, \bar{b}^*, \tilde{b}^*))$

Subroutine 1 `ComputeBounds`$(N, D(N))$

1: Compute LB with Lagrangean relaxation algorithm by solving MILP(μ)
2: Compute UB with the Benders decomposition algorithm by iteratively solving (MP) and (SP). Include the strong duality constraint $\left(e^\top - w^\top D\right) y = z^\top (k - Nx)$ to (MP) if valid for the problem. Warm-start with the optimal solution from Step 1 as the initial starting point and lower bound for (MP) if it exists
3: **return** (UB, LB)

Subroutine 2 `SolveAndBound`$(N, D(N), Incumbent)$

1: $(UB, LB) :=$ `ComputeBounds`$(N, D(N))$
2: **if** $UB < Incumbent$ **then**
3: $\quad Incumbent := UB$
4: $\quad$ Record $Incumbent$ and the optimal solution $(x^*, y^*, z^*, \bar{b}^*, \tilde{b}^*)$ of BD.
5: **end if**
6: **if** $\left|\frac{UB-LB}{LB}\right| > TOL$ **and** $LB < Incumbent$ **then**
7: $\quad Decision :=$ BRANCHING
8: **else**
9: $\quad Decision :=$ FATHOMING
10: **end if**
11: **return** $(Decision, Incumbent)$

Subroutine 3 `DecomposeDomain`$(N, D(N))$

$Count := 0$
$Branched :=$ **false**
while not $Branched$ **and** $Count < CountLimit$ **do**
  Get two sample points s_1 and s_2 from $D(N)$ and compute their associated $\left(\lambda_i^\top, \bar{b}_i^\top, \tilde{b}_i^\top\right)$ values $(i = 1, 2)$. The two sample points can be chosen randomly within $D(N)$ for instance by solving the problems $\{x|\min_x c_0 x \text{ s.t.} x \in D(N)\}$ and $\{x|\max_x c_0 x \text{ s.t.} x \in D(N)\}$, where c_0 is a random cost vector
  Compute their intersection hyperplane as $\alpha(s_1) + \lambda_1^\top(x - s_1) = \alpha(s_2) + \lambda_2^\top(x - s_2)$
  $D(N{+}1) := D(N) \cap \left\{x | \alpha(s_1) + \lambda_1^\top(x - s_1) \leq \alpha(s_2) + \lambda_2^\top(x - s_2)\right\}$ and $D(N{+}2) := D(N) \cap \left\{x | \alpha(s_1) + \lambda_1^\top(x - s_1) \geq \alpha(s_2) + \lambda_2^\top(x - s_2) + TOL\right\}$
  if $D(N{+}1) \neq \emptyset$ **and** $D(N{+}2) \neq \emptyset$ **then**
    $Branched :=$ **true**
  else
    $Count{+} = 1$
  end if
end while
if $Branched =$ **false then**
  Select arbitrary intersecting hyperplane within $D(N)$ and define $D(N{+}1)$ and $D(N{+}2)$ accordingly
end if
return $(N{+}1, N{+}2, D(N{+}1), D(N{+}2))$

2.5 Convergence of the Dynamic Algorithm

To prove that the algorithm in the previous section converges, we make the following assumptions:

Assumption 1 (A1) The feasible region of (MP), $X = \{x \in Z^{n_x} | Qx \leq q\}$, is a bounded, non-empty set.

Assumption 2 (A2) The feasible region of (SP) for a given x, $\Omega_{\text{SP}}(x) \neq \emptyset$ when $x \in X$.

Theorem 1. *Suppose that assumptions (A1) and (A2) hold and we are able to solve all sub problems within tolerance TOL. Then, the above dynamic DC-MPEC algorithm converges to a global optimum of problem (MILP), within the accuracy TOL, in a finite number of iterations.*

Proof. This theorem is proved in two steps. First, we prove that we cannot prune the subdomain containing the optimal solution. Next, we prove that the algorithm in a finite number of iterations will be able to partition in such a way that the optimal solution is in a subdomain with convex $\alpha(x)$.

1. We prune by bound when the solution of the Lagrangean subproblem $z_{\text{MILP}(\mu)} \leq Incumbent$. Geoffrion (1974) proves in Theorem 1a) that the Lagrangean subproblem is a valid lower bound for (MILP) which means no optimal solution can be lost by pruning.

2. Since we have a limit on the number of Lagrange iterations, the computation of lower bound from (MILP(μ)) ends in a finite number of iterations. The computation of the upper bound $z_{\text{B-MILP}}$ using Benders decomposition also terminates in a finite number of steps according to Benders (1962). By assumption (A1) there are a finite number of points in the domain X and each branching is forced to leave at least one point in each subdomain, which means branch-and-bound can reach subdomains containing single points within a finite number of partitions. By definition $\alpha(x)$ is convex in subdomains containing a single point.

From the following three observations, we know that the algorithm will find the global optimal solution: (i) the subdomain containing the global optimal solution cannot be pruned by 1.; (ii) we can find the subdomain containing the global optimal solution where $\alpha(x)$ is convex by 2.; and (iii) according to Benders (1962), Benders decomposition algorithm provides the global solution for a convex $\alpha(x)$. □

2.6 Scenario Decomposition

In the setting where the lower-level is a two-stage stochastic complementarity program the structure of the problem can be utilized when solving the Lagrangean subproblem (MILP(μ)) using scenario decomposition (Carøe & Schultz 1999).

The uncertainty is described by a set of scenarios j with the probability p_j, $j = 1, \ldots, J$, where $\sum_{j=1}^{J} p_j = 1$. For a problem with a two-stage stochastic program in the lower-level, Problem (1) can be written as:

$$\begin{aligned} \min_{x,y} \; & \sum_{j=1}^{J} p_j \left(c^\top x_j + \breve{d}^\top \breve{y}_j + \hat{d}_j^\top \hat{y}_j \right) \\ s.t. \; & Qx_j \leq q \quad \text{for } j = 1, \ldots, J \\ & \breve{Q} x_j \leq \breve{q} \quad \text{for } j = 1, \ldots, J \\ & A_j x_j + \breve{B}_j \breve{y}_j + \hat{B}_j \hat{y}_j \geq a_j \quad \text{for } j = 1, \ldots, J \\ & x_j = x_{j-1} \quad \text{for } j = 2, \ldots, J \\ & (\breve{y}_1, \ldots, \breve{y}_J, \hat{y}_1, \ldots, \hat{y}_J) \in S(x_1, \ldots, x_J) \end{aligned} \tag{8}$$

where $x_j \in Z^{n_x}$, $\breve{y}_j \in R^{n_{\breve{y}}}$ and $\hat{y}_j \in R^{n_{\hat{y}}}$ for $j = 1, \ldots, J$ and

$$S(x_1, \ldots, x_J) = \left\{ \begin{array}{r|r} & 0 \leq \breve{E}\breve{y}_j + \breve{e} - M^\top \breve{z}_j - \sum_{j=1}^{J} \breve{M}^\top \hat{z}_j \perp \breve{y}_j \geq 0 \\ & 0 \leq p_j \hat{E}_j \hat{y}_j + p_j \hat{e}_j - \hat{M}_j^\top \hat{z}_j - D_j^\top \hat{w}_j \perp \hat{y}_j \geq 0 \\ (\breve{y}_1, \ldots, \breve{y}_J, & 0 \leq k - Nx_j - M^\top \breve{y}_j \perp \breve{z} \geq 0 \\ \hat{y}_1, \ldots, \hat{y}_J, & 0 \leq M^\top \breve{y}_j - \hat{M}_j^\top \hat{y}_j - \hat{k}_j + \hat{N}_j x_j \perp \hat{z}_j \geq 0 \\ \breve{z}_1, \ldots, \breve{z}_J, & D\hat{y}_j + Fx_j = g \\ \hat{z}_1, \ldots, \hat{z}_J, & \text{for } j = 1, \ldots, J \\ \hat{w}_1, \ldots, \hat{w}_J) & \breve{y}_j = \breve{y}_{j-1} \\ & \breve{z}_j = \breve{z}_{j-1} \\ & \text{for } j = 2, \ldots, J \end{array} \right\} \tag{9}$$

Here $\breve{y}_j$ and $\breve{z}_j$ are the first-stage decisions and $\hat{y}_j$, $\hat{z}_j$ and $\hat{w}_j$ are the second-stage decisions of the lower-level for scenario j. The two last equation sets of $S(x_1, \ldots, x_J)$ are non-anticipativity constraints (Rockafellar & Wets 1976) included to make sure the first-stage decisions are equal in all scenarios.

We transform the complementarity conditions of Problem (9) into disjunctive constraints as described earlier, transforming the problem into a

MILP. The underlying stochastic program gives the resulting MILP matrix a structure of nearly separable blocks for each scenario. To achieve separability in the scenarios the non-anticipativity constraints are dualized using Lagrangean relaxation. Since one set of optimality conditions contain a sum over all scenarios also these constraints need to be dualized to achieve separability. Note that also the upper-level variables and some disjunctive binary variables act as first-stage variables in this setting.

Scenario decomposition corresponds to Lagrangean relaxation of an integer program. As stated in Theorem 1 in Geoffrion (1974) this gives a lower bound on the original problem which makes it valid for our lower bounding purpose, but does not guarantee a tight bound.

3 Results

The dynamic algorithm has been tested both on randomly generated test data for the general model formulation and on three cases for an application of the DC-MPEC problem from the Norwegian natural gas supply chain. For all tests the tolerance was set to $TOL = 10^{-5}$.

3.1 Results on Randomly Generated Data

In this section, we present our computational results with the dynamic DC-MPEC algorithm based on randomly generated data. Each lower-level problem is created by deriving the KKT conditions from a LP or convex QP problem. 59 LP-based problems are generated by random data from the interval [-500,500] with a uniform distribution. Similarly, data for 40 QP-based problems are sampled from a uniform distribution [-100,100]. The test problems are grouped according to the dimension of the variable vectors and underlying problem type, as summarized in Table 4. Sensitivity analysis has been conducted to find the working range for the disjunctive constant C, and the values reported in Table 4 is within this range and correponds to the results reported in the following tables. Data for Groups 1 to 4 is provided in the online appendix for Gabriel et al. (2010), while the rest are new test problems. All tests were conducted on a computer with 2.34 GHz processor and 23.55 GB memory. The algorithm was implemented with MATLAB (ver. 7.0) and GAMS (ver. 23.6) interfacing where GAMS used Xpress-MP for its MILP solver. Test runs using more than 10 hours were stopped, giving rise to "n/a" in the result tables.

Table 5 lists the number of subdomains and the number of sampling and branching attempts for the alternative algorithms. Results are given for the

Table 4: Random generated test problem groups. MP and SP give the dimension of the variable vector. The value of the disjunctive constant used for the tests are given by C. Whether the lower level problem is derived from a LP or QP is indicated by the problem type.

Gr	MP		SP			C	Probl
	x	type	y	z	b		type
1	2	int	5	2	7	1.E+03	LP
2	5	int	5	2	7	1.E+03	LP
3	10	bin	10	5	15	300	LP
4	10	bin	15	10	25	1.E+06	LP
5	2	int	400	200	600	1.E+06	LP
6	20	bin	100	100	200	1.E+05	LP
7	2	int	100	50	150	1.E+06	QP

original static algorithm ("Stat") as in Gabriel et al. (2010) and the dynamic algorithm with and without warm-starting the Benders algorithm with the solution from Lagrangean relaxation ("Dyn ws" and "Dyn", respectively). For the dynamic algorithm the number of subdomains corresponds to leaf nodes in the branch-and-bound tree, while in the static algorithm subdomains are identified by comparing $(\lambda^\top, \bar{b}^\top, \tilde{b}^\top)$ as described in Gabriel et al. (2010). We observe that the dynamic algorithm reduces the number of subdomains and samplings compared to the static algorithm. In Table 6 the solution times for the same test problems and algorithms are given. "Bounding time" covers the Lagrangean and Benders algorithms (Subroutines 1-2), "Sampling time" sampling and branching the x-domain (Subroutine 3), and "Total time" is the sum of the two. The dynamic algorithm shows a significant reduction in solution time compared to the static algorithm, mainly caused by a major reduction in "Sampling time". This corresponds well to the reduction in number of subdomains and sampling, and shows that the gain from fewer subdomains because of dynamic branching is larger than the added work on computing bounds. We observe that including the warm-start of the Benders algorithm makes the dynamic algorithm able to solve the problem in the root node for all test problems, also those with non-convex $\alpha(x)$, which indicates that Lagrangean relaxation gives a very strong lower bound for the problem. We have not been able to prove that this result is guaranteed for the problem class in general. The tables also show that the dynamic algorithm with warm-start solves all test problems, while the dynamic algorithm without warm-start leaves two problems unsolved and the static leaves five unsolved due to long solution times.

Table 5: Number of subdomains and samplings for random generated test problem solved by static algorithm and dynamic algorithm without and with warm-start (ws). # subdomains referes to the finest partitioning of the x-domain, which corresponds to the leaf nodes of the branch-and-bound tree for the dynamic algorithm. For the static algorithm the subdomains are identified through sampling and comparing $(\lambda^\top, \bar{b}^\top, \tilde{b}^\top)$ as described in Gabriel et al. (2010). The true number of subdomains are identified by enumeration, where a subdomain is defined as a set of integer x points where $(\lambda^\top, \bar{b}^\top, \tilde{b}^\top)$ is the same. Mean and median values cover all test problems that were solved by all three algorithms.

Gr	ID	# subdomains				# samplings		
		True	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
mean		19.5	19.7	3.9	1	40.6	58.5	0
median		14	4	1	1	4.5	2.5	0
1	1	4	1	1	1	2	0	0
1	2	4	1	1	1	2	0	0
1	3	6	5	2	1	10	4	0
1	4	5	4	1	1	2	2	0
2	1	4	1	1	1	2	8	0
2	2	4	1	1	1	2	0	0
2	3	6	1	1	1	2	0	0
2	4	7	198	1	1	382	0	0
3	1	6	1	3	1	2	24	0
3	2	16	47	11	1	123	139	0
3	3	10	61	4	1	195	37	0
3	4	20	29	1	1	71	3	0
3	5	20	50	1	1	115	14	0
3	6	7	17	17	1	40	446	0
3	7	14	1	3	1	2	100	0
3	8	8	16	1	1	31	17	0
3	9	22	26	1	1	74	7	0
3	10	9	1	1	1	2	0	0
3	11	9	4	3	1	4	100	0
3	12	10	1	2	1	2	22	0
3	13	16	11	3	1	27	92	0
3	14	14	77	2	1	174	136	0
3	15	14	1	1	1	2	5	0
3	16	33	65	5	1	188	62	0
3	17	32	63	1	1	168	2	0
3	18	20	22	11	1	26	247	0

Table 5: **continued**

Gr	ID	# subdomains				# samplings		
		True	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
4	1	38	4	13	1	15	149	0
4	2	41	1	35	1	1	708	0
4	3	27	1	29	1	1	831	0
4	4	12	99	35	1	223	395	0
4	5	52	2	1	1	1	6	0
4	6	6	12	13	1	48	502	0
4	7	13	20	52	1	39	552	0
4	8	60	105	3	1	259	70	0
4	9	7	1	2	1	2	58	0
4	10	33	41	5	1	97	33	0
4	11	24	14	5	1	12	138	0
4	12	15	24	3	1	47	0	0
4	13	28	79	5	1	206	36	0
5	1	16	19	2	1	37	2	0
5	2	6	12	2	1	23	2	0
5	3	6	1	1	1	2	6	0
5	4	11	2	1	1	3	0	0
5	5	42	33	2	1	78	2	0
5	6	92	22	1	1	43	0	0
5	7	21	9	1	1	23	0	0
5	8	42	26	1	1	51	0	0
5	9	21	3	1	1	5	0	0
5	10	23	24	1	1	49	0	0
6	1	n/a	1	1	1	2	0	0
6	2	n/a	3	1	1	5	5	0
6	3	n/a	2	1	1	3	0	0
6	4	n/a	1	1	1	2	0	0
6	5	n/a	1	1	1	2	0	0
6	6	n/a	1	1	1	2	0	0
6	7	n/a	1	1	1	2	0	0
6	8	n/a	1	1	1	2	0	0
6	9	n/a	2	1	1	3	0	0
6	10	n/a	n/a	n/a	1	n/a	n/a	0

Table 5: **continued**

Gr	ID	# subdomains				# samplings		
		True	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
7	1	n/a	3	1	1	5	0	0
7	2	n/a	5	1	1	9	2	0
7	3	n/a	1	3	1	4	59	0
7	4	n/a	2	5	1	3	62	0
7	5	n/a	10	5	1	19	41	0
7	6	n/a	7	1	1	19	0	0
7	7	n/a	1	1	1	2	0	0
7	8	n/a	1	3	1	2	34	0
7	9	n/a	1	1	1	2	0	0
7	10	n/a	1	1	1	2	2	0
7	11	n/a	2	2	1	3	6	0
7	12	n/a	1	1	1	2	0	0
7	13	n/a	2	2	1	4	20	0
7	14	n/a	1	1	1	2	0	0
7	15	n/a	1	1	1	2	0	0
7	16	n/a	3	1	1	3	0	0
7	17	n/a	2	1	1	3	0	0
7	18	n/a	1	1	1	2	0	0
7	19	n/a	26	1	1	34	0	0
7	20	n/a	37	3	1	55	17	0
7	21	n/a	5	1	1	7	2	0
7	22	n/a	40	2	1	121	44	0
7	23	n/a	12	3	1	87	58	0
7	24	n/a	1	1	1	2	0	0
7	25	n/a	1	1	1	2	0	0
7	26	n/a	1	1	1	2	0	0
7	27	n/a	10	4	1	27	45	0
7	28	n/a	68	1	1	313	0	0
7	29	n/a	53	1	1	111	4	0
7	30	n/a	53	1	1	116	0	0
7	31	n/a	72	2	1	147	36	0
7	32	n/a	12	4	1	30	47	0
7	33	n/a	65	2	1	152	50	0
7	34	n/a	5	4	1	10	9	0
7	35	n/a	75	1	1	156	0	0
7	36	n/a	1	1	1	2	0	0
7	37	n/a	n/a	n/a	1	n/a	n/a	0
7	38	n/a	n/a	1	1	n/a	0	0
7	39	n/a	n/a	1	1	n/a	0	0
7	40	n/a	n/a	1	1	n/a	0	0

Table 6: Solution times for random generated test problem solved by static algorithm and dynamic algorithm without and with warm-start (ws). The convexity of $\alpha(x)$ is checked through plotting, and is therefore only available for 2-dimensional problems. Bounding time covers Benders and Lagrangean algorithm, Sampling time is time for sampling and branching while Total time is the sum Bounding and Sampling. Mean and median values covers all test problems that were solved by all three algorithms. All times in seconds.

Gr	ID	Convex	Total time			Sampling time			Bounding time		
		$\alpha(x)$?	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
mean			614.03	296.23	42.34	505.43	110.83	0	108.60	185.40	42.34
median			113.78	61.37	2.64	85.57	10.47	0	36.85	37.93	2.64
1	1	Yes	3.65	0.86	0.78	2.62	0	0	1.03	0.86	0.78
1	2	Yes	3.08	0.84	0.78	2.38	0	0	0.7	0.84	0.78
1	3	No	53.83	7.15	0.71	45.98	4.45	0	7.85	2.71	0.71
1	4	No	21.97	3.74	0.78	13.72	2.22	0	8.25	1.52	0.78
2	1	n/a	3.69	18.31	0.81	2.36	9.00	0	1.33	9.31	0.81
2	2	n/a	3.23	6.41	0.80	2.24	0	0	0.99	6.41	0.80
2	3	n/a	3.25	3.65	0.73	2.26	0	0	0.99	3.65	0.73
2	4	n/a	4 372.40	10.95	0.79	3 461.7	0	0	910.70	10.95	0.79
3	1	n/a	3.72	66.07	0.78	2.59	29.01	0	1.12	37.06	0.78
3	2	n/a	602.74	308.42	0.83	526.43	167.58	0	76.31	140.84	0.83
3	3	n/a	949.43	80.54	0.86	856.59	43.99	0	92.84	36.55	0.86
3	4	n/a	357.67	6.72	0.78	308.12	3.66	0	49.55	3.06	0.78
3	5	n/a	589.89	30.83	0.84	502.13	16.70	0	87.76	14.13	0.84
3	6	n/a	204.17	916.14	0.79	178.06	543.31	0	26.11	372.83	0.79
3	7	n/a	3.67	268.67	0.84	2.54	127.19	0	1.13	141.48	0.84
3	8	n/a	191.79	39.64	0.77	146.61	21.01	0	45.18	18.63	0.77
3	9	n/a	373.93	14.21	0.85	325.22	8.64	0	48.71	5.57	0.85

Table 6: **continued**

Gr	ID	Convex	Total time			Sampling time			Bounding time		
		$\alpha(x)$?	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
3	10	n/a	2.99	1.31	0.85	2.58	0	0	0.40	1.31	0.85
3	11	n/a	34.60	258.38	1.04	21.72	125.08	0	12.88	133.30	1.04
3	12	n/a	4.78	59.84	0.78	2.51	27.33	0	2.26	32.51	0.78
3	13	n/a	135.42	204.19	0.89	116.71	112.34	0	18.71	91.85	0.89
3	14	n/a	938.08	314.31	0.84	784.07	171.50	0	154.01	142.81	0.84
3	15	n/a	2.93	11.29	0.82	2.52	6.07	0	0.41	5.22	0.82
3	16	n/a	938.34	137.76	0.76	824.76	76.34	0	113.58	61.43	0.76
3	17	n/a	862.79	5.93	0.83	733.41	2.50	0	129.38	3.43	0.83
3	18	n/a	223.64	593.27	0.91	152.38	303.82	0	71.26	289.45	0.91
4	1	n/a	74.55	370.99	0.93	67.16	195.94	0	7.39	175.05	0.93
4	2	n/a	2.8	2 286.52	1.69	1.93	891.61	0	0.87	1 394.91	1.69
4	3	n/a	2.68	2 239.25	1.69	1.74	1 060.1	0	0.94	1 179.15	1.69
4	4	n/a	1 276.04	1 010.31	1.18	1 074.4	513.2	0	201.64	497.11	1.18
4	5	n/a	13.19	20.14	0.88	4.86	7.45	0	8.33	12.69	0.88
4	6	n/a	250.85	1 100.45	0.81	213.46	609.88	0	37.39	490.57	0.81
4	7	n/a	231.46	1 567.46	0.87	180.94	703.81	0	50.52	863.65	0.87
4	8	n/a	1 366.17	269.18	1.8	1 158.9	90.51	0	207.27	178.67	1.8
4	9	n/a	16.21	179.64	1.02	8.53	73.59	0	7.68	106.05	1.02
4	10	n/a	514.46	125.42	1.92	427.97	43.06	0	86.49	82.37	1.92
4	11	n/a	227.28	295.99	0.97	128.81	176.47	0	98.47	119.52	0.97
4	12	n/a	284.97	2.53	0.81	221.23	0	0	63.74	2.53	0.81
4	13	n/a	1 105.88	123.33	2.12	954.53	44.86	0	151.35	78.48	2.12

Table 6: **continued**

Gr	ID	Convex	Total time			Sampling time			Bounding time		
		$\alpha(x)$?	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
5	1	No	1 985.52	251.12	30.47	1 717.40	157.75	0	268.12	93.37	30.47
5	2	Yes	1 372.76	294.16	30.22	1 077.00	149.88	0	295.76	144.28	30.22
5	3	No	104	669.86	30.40	92.24	461.78	0	11.76	208.08	30.40
5	4	Yes	231.12	148.27	30.56	137.98	0	0	93.14	148.27	30.56
5	5	No	4 307.94	596.21	31.96	3 806.40	156.53	0	501.54	439.68	31.96
5	6	Yes	2 418.82	134.59	30.7	2 094.50	0	0	324.32	134.59	30.70
5	7	No	1 339.73	180.39	34.50	1 083.8	0	0	255.93	180.39	34.5
5	8	Yes	2 878.06	128.22	30.47	2 496.20	0	0	381.86	128.22	30.47
5	9	Yes	365	165.89	30.37	226.44	0	0	138.56	165.89	30.37
5	10	No	2 704.98	33.52	32.36	2 373.90	0	0	331.08	33.52	32.36
6	1	n/a	58.17	42.82	4.11	16.6	0	0	41.57	42.82	4.11
6	2	n/a	87.13	170.17	4.28	41.59	42.58	0	45.54	127.59	4.28
6	3	n/a	115.3	58.37	4.08	24.63	0	0	90.67	58.37	4.08
6	4	n/a	52.64	38.32	4.26	16.37	0	0	36.27	38.32	4.26
6	5	n/a	39.93	30.87	4.31	16.61	0	0	23.32	30.87	4.31
6	6	n/a	52.66	45.05	4.33	16.35	0	0	36.32	45.05	4.33
6	7	n/a	52.44	37.55	3.99	16.4	0	0	36.04	37.55	3.99
6	8	n/a	41.94	29.72	4.13	16.33	0	0	25.62	29.72	4.13
6	9	n/a	65.5	24.91	4.04	24.66	0	0	40.84	24.91	4.04
6	10	n/a	n/a	n/a	3.97	n/a	n/a	0	n/a	n/a	3.97

Table 6: **continued**

Gr	ID	Convex	Total time			Sampling time			Bounding time		
		$\alpha(x)$?	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
7	1	Yes	112.27	12.05	2.99	78.89	0	0	33.38	12.05	2.99
7	2	Yes	169.52	30.07	2.95	135.99	12.09	0	33.53	17.98	2.95
7	3	Yes	75.91	761.02	2.94	62.47	374.09	0	13.44	386.93	2.94
7	4	Yes	66.31	769.70	2.95	45.27	381.47	0	21.05	388.23	2.95
7	5	Yes	358.20	542.91	3.06	288.53	245.89	0	69.67	297.02	3.06
7	6	Yes	402.86	16.17	3.43	318.02	0	0	84.84	16.17	3.43
7	7	Yes	26.45	13.22	3.22	14.33	0	0	12.13	13.22	3.22
7	8	Yes	39.64	777.80	3.15	23.77	325.77	0	15.87	452.03	3.15
7	9	Yes	36.30	15.65	3.26	21.28	0	0	15.02	15.65	3.26
7	10	Yes	30.37	50.27	3.21	19.10	21.21	0	11.27	29.06	3.21
7	11	Yes	33.56	66.52	2.60	15.92	31.87	0	17.64	34.66	2.60
7	12	Yes	22.31	13.30	2.73	10.80	0	0	11.51	13.30	2.73
7	13	Yes	35.92	215.91	2.60	21.41	106.08	0	14.51	109.83	2.60
7	14	Yes	22.36	10.97	2.58	10.78	0	0	11.58	10.97	2.58
7	15	Yes	26.65	16.85	2.60	11.14	0	0	15.50	16.85	2.60
7	16	Yes	35.95	15.63	2.63	21.33	0	0	14.62	15.63	2.63
7	17	Yes	34.25	14.31	2.69	16.12	0	0	18.13	14.31	2.69
7	18	Yes	22.33	12.61	2.61	10.80	0	0	11.53	12.61	2.61
7	19	Yes	335.77	15.05	2.70	182.85	0	0	152.92	15.05	2.70
7	20	Yes	404.77	170.31	2.54	294.86	90.35	0	109.91	79.96	2.54

Table 6: **continued**

Gr	ID	Convex	Total time			Sampling time			Bounding time		
		$\alpha(x)$?	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws	Stat	Dyn	Dyn ws
7	21	Yes	69.07	35.48	3.84	40.93	11.93	0	28.14	23.54	3.84
7	22	Yes	2 222.41	454.72	3.65	1 986.00	218.75	0	236.41	235.97	3.65
7	23	Yes	2 282.70	691.04	3.07	2 147.00	335.19	0	135.70	355.85	3.07
7	24	Yes	22.19	12.47	3.75	11.58	0	0	10.61	12.47	3.75
7	25	Yes	26	15.87	3.62	11.74	0	0	14.26	15.87	3.62
7	26	Yes	25.88	15.87	3.12	11.69	0	0	14.20	15.87	3.12
7	27	Yes	688.69	521.51	3.66	600.03	275.15	0	88.66	246.36	3.66
7	28	Yes	3 073.97	16.16	3.9	2 887.40	0	0	186.57	16.16	3.90
7	29	Yes	1 015.68	62.91	3.59	874.39	27.78	0	141.29	35.13	3.59
7	30	Yes	1 039.43	14.00	3.74	914.59	0	0	124.84	14.00	3.74
7	31	Yes	2 708.87	655.22	2.64	2 312.60	193.22	0	396.27	462.00	2.64
7	32	Yes	550.99	435.16	2.65	433.41	249.03	0	117.58	186.13	2.65
7	33	Yes	2 748.08	752.89	2.87	2 423.50	289.52	0	324.58	463.37	2.87
7	34	Yes	70.94	113.12	2.69	59.45	47.91	0	11.50	65.21	2.69
7	35	Yes	1 082.78	11.66	2.54	942.78	0	0	140.00	11.66	2.54
7	36	No	3 273.79	4 456.41	3 485.45	1 487.50	0	0	1 786.29	4 456.41	3 485.45
7	37	No	n/a	n/a	6 141.19	n/a	n/a	0	n/a	n/a	6 141.19
7	38	No	n/a	6 134.12	6 134.52	n/a	0	0	n/a	6 134.12	6 134.52
7	39	No	n/a	3 372.82	2 407.15	n/a	0	0	n/a	3 372.82	2 407.15
7	40	No	n/a	2 414.44	1 801.42	n/a	0	0	n/a	2 414.44	1 801.42

3.2 Application to a Natural Gas Supply Chain

3.2.1 Background

The presented algorithm is tested on a stochastic two-stage problem from the Norwegian natural gas supply chain. For this problem we use the scenario decomposition described earlier. Because of high investment costs, thorough planning is required for field and infrastructure developments in the natural gas industry. This planning typically has a bilevel structure where the investment level should be coordinated within the network, taking into account the competitive behavior of multiple producers in the operational level. Further, the investments are binary decisions taken under both long-term and operational uncertainty.

3.2.2 Model Description

The model presented here is a MPEC that can be written in this form:

$$\begin{aligned} &\min_{x,y} c^\top x + d^\top y \\ &s.t.\, Qx \leq q \\ &\quad y \in S(x) \end{aligned} \tag{10}$$

where $x \in Z^{n_x}$, $y \in R^{n_y}$, and

$$S(x) = \left\{ y \;\middle|\; \begin{array}{r} 0 \leq z \perp My + Nx - k \geq 0 \\ 0 \leq y \perp Ey + e - M^\top z \geq 0 \\ Ax + By \geq a \end{array} \right\} \tag{11}$$

The upper-level takes the perspective of a central planner making binary decisions on which fields and pipelines to invest in and when to invest. These decisions are represented by x. The investments have investment costs, c, and there can be dependencies between the possible investments given by $Qx \leq q$. The central planner maximizes the long term profits generated in the supply chain taking into account the short term operations, y.

For the lower-level problem we use a multi-period version of the model by Midthun et al. (2007) with some extension to facilitate a connection to the upper-level decisions. Midthun et al. (2007) describes a stochastic mixed complementarity problem where several producers make production decisions, trade in a transportation market, deliver natural gas in long term contracts and sell natural gas on the spot market. The producers maximize their profit consisting of natural gas sales income, transportation costs and production cost. The natural gas spot market has exogenously given

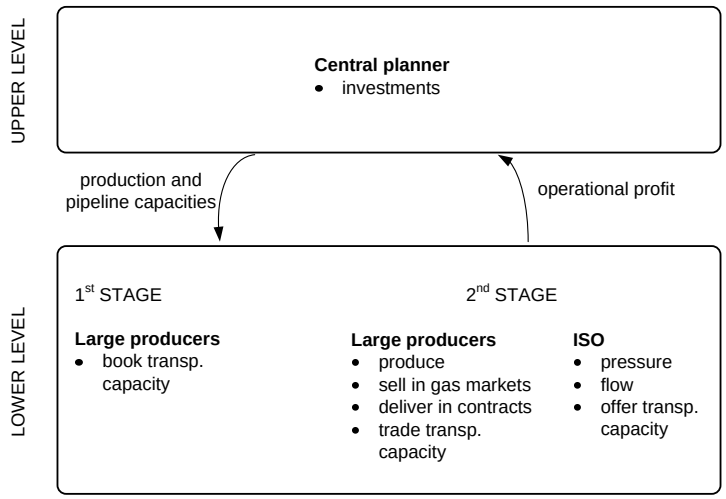

Figure 4: Overview over natural gas value chain model

stochastic prices and delivery obligations in contracts are stochastic. The independent system operator (ISO) routes the gas and makes transportation capacity available in the transportation market according to the physical capacities of the network. The market for transportation capacity is modeled endogenously in two stages. In the primary market large prequalified producers book capacity at a fixed price before the uncertainty is resolved. In the secondary market the ISO offers spare capacity, the large producers trade and the optimality conditions of a competitive fringe give a demand curve that clears the market. One producer will not know the other producers' primary booking in the second stage, an assumption that makes it possible to formulate the whole operational level as a Generalized Nash equilibrium. Figure 4 gives an overview of this model.

Midthun et al. (2007) shows that a competitive transportation market where decisions partly are taken under uncertainty results in some inefficiency compared to centralized coordination maximizing the social surplus of the supply chain. These losses are caused by excess transportation booking that carries forward to excess production and sale. The invesment decisions by a centralized upper level planner in our model will typically search for network structures that counteract these losses by providing a flexible network design.

3.2.3 Numerical Results

Three small instances of this problem are solved by the dynamic DC-MPEC algorithm. Two datasets (Datasets 1 and 2) are entirely synthetic, while the last (Dataset 3) has production capacities from an extract of the existing

fields. Each dataset have several instances with different number of scenarios.

For the natural gas application, the main coordination of the dynamic DC-MPEC algorithm is implemented in Matlab. Scenario decomposition was used for lower bounding, implemented with Mosel/Xpress to solve subproblems. To speed up the upper bounding the mixed complementarity problem from the lower-level was solved by GAMS/PATH and provided an initial solution for the binary variables of the Benders subproblem solved with GAMS/Xpress. As benchmark the whole bilevel problem is formulated as a single MILP and solved with Mosel/Xpress.

Table 7 presents the datasets and results from the tests. The disjunctive constant C was 10^5 for Dataset 1 and 10^6 for Dataset 2 and 3. Dataset 1 and 2 were tested on a HP dl160 G3 with 2x3.0GHz Intel E5472 Xeon processors and 16Gb RAM and Dataset 3 was tested on a Pentium 4 with 3.6GHz processor and 3.0Gb RAM.

Table 7: Numerical results for natural gas application. Three datasets are tested, each for several different numbers of scenarios. MP and SP give the dimention of the variable vectors. For the dynamic algorithm with warm-start the upper and lower bound and solution time in seconds is reported. For the benchmark, Xpress solving (MILP) is the objective value and solution time in seconds reported.

Dataset						Dynamic with warm-start			Benchmark	
	#	MP	SP			UB	LB	Time	Obj	Time
	scen	x	y	z	b				value	
1	10	12	324	604	688	1.09832E6	1.09832E6	11.2	1.09832E6	2.1
1	15	12	484	904	1 028	1.11765E6	1.11765E6	12.9	1.11765E6	2.2
1	20	12	644	1 204	1 368	1.19232E6	1.19232E6	14.4	1.19232E6	2.2
1	25	12	804	1 504	1 708	1.22435E6	1.22435E6	16.7	n/a[a]	>2h
1	30	12	964	1 804	2 048	1.26057E6	1.26057E6	17.5	n/a	n/a
1	100	12	3 204	6 004	6 808	1.20326E6	1.20326E6	50.4	n/a	n/a
1	500	12	16 004	30 004	34 008	1.14888E6	1.14888E6	472.2	n/a	n/a
1	1 000	12	32 004	60 004	68 008	1.14710E6	1.14710E6	7454.6	n/a	n/a

[a]The benchmark code did not complete it running in 2 hours, so we stopped it.

Table 7: **continued**

Dataset						Dynamic with warm-start			Benchmark	
	#	MP	SP			UB	LB	Time	Obj	Time
	scen	x	y	z	b				value	
2	10	6	762	1 392	1 644	-4.41152	-4.41152	16.9	-4.41152	1.1
2	25	6	1 887	3 462	4 074	-4.50951	-4.50951	39.4	-4.50951	3.2
2	50	6	3 762	6 912	8 124	-4.57339	-4.57339	115.7	-4.57339	6.0
2	75	6	5 637	10 362	12 174	-4.48222	-4.48222	170.8	-4.48222	11.4
2	100	6	7 512	13 812	16 224	-4.53883	-4.53883	336.2	-4.53883	16.9
2	300	6	22 512	41 412	48 624	-4.50862	-4.50862	4589.0	-4.50862	128.0
2	500	6	37 512	69 012	81 024	-4.50097	-4.50097	17114.4	-4.50097	414.5
2	700	6	52 512	96 612	113424	-4.49860	-4.49861	27201.4	-4.49861	647.0
3	10	6	762	1 392	1 644	-0.819261	-0.819261	44.6	-0.819261	2.7
3	25	6	1 887	3 462	4074	-0.821139	-0.821139	87.8	-0.821139	36.0
3	50	6	3 762	6 912	8 124	-0.766766	-0.766766	234.1	n/a[b]	>23h
3	75	6	5637	10 362	12 174	-0.784225	-0.784223	508.6	n/a	n/a
3	100	6	7 512	13 812	16 224	-0.783515	-0.783515	707.7	n/a	n/a

[b]The benchmark code did not complete it running in 23 hours, so we stopped it. Until then, best solution and best bound found were 0.765686 and 0.766766, respectively.

The benchmark was faster in the small instances, but our algorithm was able to solve substantially larger instances for Dataset 1 and 3. It is not surprising to find the benchmark faster on small instances, since our research code combining several software has a significant overhead in data transfer. We observe that in all tests where the real optimal value was known from the benchmark, the dynamic DC-MPEC algorithm provided an optimal solution, as expected based on the theory.

We have experienced challenges related to numerical stability in the testing of our algorithm. In several occasions the solver provided a solution to a MILP problem, but when asking the solver to fix the binary variables according to the solution and resolve, the solver claimed the resulting LP to be infeasible. To overcome this situation the tests on Dataset 2 and 3 were run without the presolve function activated in the Xpress solver (www.gams.com/dd/docs/solvers/xpress.pdf) for upper bounding, and on Dataset 3 we also had to use different values for MIP tolerance on different instances. Naturally, deactivating presolve gives a disadvantage to the dynamic DC-MPEC algorithm when it comes to solution time.

4 Conclusions

We have presented a dynamic DC-MPEC algorithm to solve discretely constrained mathematical programs with equilibrium constraints (DC-MPEC) which is a class of bilevel program with integer program in the upper-level and mixed complementary problem in the lower-level. We develop a new branch-and-bound method for DC-MPEC problems applying Benders decomposition and Lagrangean relaxation methods. We provide convergence theory for the new method showing that it will find the global optimum and implement the new dynamic DC-MPEC algorithm on a set of test problems for both convex and non-convex domains. The numerical results show that the dynamic DC-MPEC algorithm outperforms the static counterpart presented by Gabriel et al. (2010) due to reduced sampling and branching efforts. The dynamic algorithm is further improved by warm-starting the Benders algorithm with the solution found by Lagrangean relaxation. We enhance the new method with the scenario decomposition method (Carøe & Schultz 1999) for two-stage stochastic DC-MPEC problems with discrete probability space. Then we compare the stochastic DC-MPEC algorithm with the single level approach by Fortuny-Amat & McCarl (1981) for a application for the Norwegian natural gas value chain. These numerical results present the effectiveness of our branch-and-bound algorithm and demonstrate the potential of the algorithm for a decision support tool for upper-level planners whose

decisions are discrete.

Acknowledgements

The project is partially supported by the Research Council of Norway under grant 175967/S30.

*Appendix

A Theorem

Theorem 2. *The solution of (SP) from the Benders decomposition algorithm provides an upper bound on* $z_{\text{MILP}} = d^\top x + d^\top y$ *for any given partition, and the optimal solution* $z_{\text{MILP}} = d^\top x^* + d^\top y^*$ *for any partition where* $\alpha(x)$ *is convex.*

Proof. By the assumption (A2) any $x^{(v)}$ which is a feasible in (MP) is also feasible in (MILP). For a given $x^{(v)}$ the feasible region of (SP) is identical to the feasible region for the variables $y, z, \bar{b}$ and $\tilde{b}$ of (MILP), which makes a solution of (SP) feasible in (MILP). And since the function $z^{\text{up}}\left(x^{(v)}\right)$ is identical to the objective function of (MILP), it is an upper bound of (MILP). Benders (1962) proves that Benders decomposition algorithm converges to the optimal solution in the case of a convex $\alpha(x)$. □

References

Audet, C., Savard, G. & Zghal, W. (2007), 'New branch-and-cut algorithm for bilevel linear programming', *Journal of Optimization Theory and Applications* **134**, 353–370.

Bard, J. F. & Moore, J. T. (1990), 'A branch and bound algorithm for the bilevel programming problem', *SIAM Journal on Scientific and Statistical Computations* **11**(2), 281–292.

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

Carøe, C. C. & Schultz, R. (1999), 'Dual decomposition in stochastic integer programming', *Operations Research Letters* **24**, 37–45.

Colson, B., Marcotte, P. & Savard, G. (2007), 'An overview of bilevel optimization', *Annals of Operations Research* **153**, 235–256.

Conejo, A. J., Castillo, E., Minguez, R. & Garca-Bertrand, R. (2006), *Decomposition Techniques in Mathematical Programming: Engineering and Science Application*, Springer. ISBN: ISBN: 978-3-540-27685-2.

Dempe, S. (2002), *Foundations of Bilevel Programming*, Kluwer Academic Publisher.
URL: *http://www.springerlink.com/content/p3n840/*

DeNegre, S. T. & Ralphs, T. K. (2009), 'A branch-and-cut algorithm for integer bilevel linear programs', *Operations Research/Computer Science Interfaces* **47**, 65–78.

Falk, J. E. (1969), 'Lagrange multipliers and nonconvex programs', *SIAM Journal on Control* **7**(4), 534–545.

Fortuny-Amat, J. & McCarl, B. (1981), 'A representation and economic interpretation of a two-level progamming problem', *The Journal of the Operational Research Society* **32**(9), 783–792.

Fukushima, M. & Lin, G.-H. (2004), Smoothing methods for mathematical programs with equilibrium constraints, *in* 'Proceedings of the ICKS'04', IEEE Computer Society, pp. 206–213.

Gabriel, S. A. & Leuthold, F. U. (2010), 'Solving discretely-constrained mpec problems with applications in electric power markets', *Energy Economics* **32**(1), 3–14.

Gabriel, S. A., Shim, Y., Conejo, A. J., de la Torre, S. & Garcia-Bertrand, R. (2010), 'A Benders decomposition method for discretely-constrained mathematical programs with equilibrium constraints with applications in energy', *The Journal of the Operational Research Society* **61**(9), 1404–1419. Paper under review, JORS, October 16, 2007.

Geoffrion, A. M. (1974), 'Lagrange relaxation for integer programming', *Mathematical Programming Study* **2**, 82–114.

Hansen, P., Jaumard, B. & Savard, G. (1992), 'New branch-and-bound rules for linear bilevel programming', *SIAM Journal on Scientific and Statistical Computing* **13**, 1194–1217.

Hu, J., Mitchell, J. E., Pang, J., Bennett, K. P. & Kunapuli, G. (2008), 'On the global solution of linear programs with linear complementarity constraints', *SIAM Journal of Optimization* **19**(1), 445–471.

Labbé, M., Marcotte, P. & Savard, G. (1998), 'A bilevel model of taxation and its application to optimal highway pricing', *Management Science* **44**(12), 1608–1622.

Luo, Z. Q., Pang, J. S. & Ralph, D. (1996), *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, Cambridge. ISBN: 0-521-57290-8.

Meng, Q., Huang, Y. & Cheu, R. L. (2009), 'Competitive facility location on decentralized supply chains', *European Journal of Operational Research* **196**, 487–499.

Meng, Q. & Wang, X. (2011), 'Intermodal hub-and-spoke network design: Incorporating multiple stakeholders and multi-type containers', *Transportation Research Part B* **45**(4), 724–742.

Mesbah, M., Sarvi, M., Ouveysi, I. & Currie, G. (2011), 'Optimization of transit priority in the transportation network using a decomposition methodology', *Transportation Research Part C* **19**, 363–373.

Midthun, K. T., Bjrndal, M., Tomasgard, A. & Smeers, Y. (2007), Capacity booking in a transportation network with stochatic demand and secondary market for transportation capacity, *in* K. T. Midthun, ed., 'Optimization models for liberalized natural gas markets', PhD thesis, NTNU, Trondheim, Norway.

Mitsos, A. (2010), 'Global solution of nonlinear mixed-integer bilevel programs', *Journal of Global Optimization* **47**(4), 557–582.

Moore, J. T. & Bard, J. F. (1990), 'The mixed integer linear bilevel programming problem', *Operations Research* **38**, 911–921.

Outrata, J., Kocvara, M. & Zowe, J. (1998), *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints: Theory, Applications and Numerical Results*, Kluwer Academic Publishers, Boston. ISBN: 978-0-7923-5170-2.

Rockafellar, R. T. & Wets, R. J.-B. (1976), 'Nonanticipativity and l^1-martingales in stochastic optimization problems', *Mathematical Programming Study* **6**, 170–187.

Saharidis, G. K., Boile, M. & Theofanis, S. (2011), 'Initialization of the Benders master problem using valid inequalities applied to fixed-charge network problems', *Expert Systems with Applications* **38**, 6627–6636.

Saharidis, G. K. & Ierapetritou, M. G. (2009), 'Resolution method for mixed bilevel linear problems based on decomposition technique', *Journal of Global Optimization* **44**, 29–51.

Saharidis, G. K. & Ierapetritou, M. G. (2010), 'Improving Benders decomposition using maximum feasible subsystem (mfs) cut generation strategy', *Computers and Chemical Engineering* **34**, 1237–1245.

van Roy, T. J. (1983), 'Cross decomposition for mixed integer programming', *Mathematical Programming* **25**, 46–63.

van Roy, T. J. (1986), 'A cross decomposition algorithm for capacitated facility location', *Operations Research* **34**(1), 145–163.

Wang, D. Z. W. & Lo, H. K. (2008), 'Multi-fleet ferry service network design with passenger preferences for differantial services', *Transportation Research Part B* **42**, 798–822.

Wen, U. P. & Yang, Y. H. (1990), 'Algorithms for solving the mixed integer two-level linear programming problem', *Computers and Operations Research* **17**(2), 133–142.