Multistage Discrete Optimization

Ted Ralphs[1] Joint work with Sahar Tahernajad[1], Scott DeNegre[3], Menal Güzelsoy[2], Anahita Hassanzadeh[4]

[1]COR@L Lab, Department of Industrial and Systems Engineering, Lehigh University [2]SAS Institute, Advanced Analytics, Operations Research R & D
[3]The Hospital for Special Surgery
[4]Climate Corp

IWOBIP, Monterrey, Mexico, 11 March 2016

Outline

Outline

A Bit of Game Theory

- Our goal is to analyze certain *finite extensive-form games*, which are sequential games involving n players.

Loose Definition

- The game is specified on a tree with each node corresponding to a move and the outgoing arcs specifying possible choices.
- The leaves of the tree have associated payoffs.
- Each player's goal is to maximize payoff.
- There may be *chance* players who play randomly according to a probability distribution and do not have payoffs (*stochastic games*).

- All players are rational and have perfect information.
- The problem faced by a player in determining the next move is a *multilevel/multistage* optimization problem.
- The move must be determined by taking into account the *responses of the other players*.

Multilevel and Multistage Games

- We use the term *multilevel* for competitive games in which there is no chance player.
- We use the term *multistage* for cooperative games in which all players receive the same payoff, but there are chance players.
- A *subgame* is the part of a game that remains after some moves have been made.

Stackelberg Game

- A Stackelberg game is a game with two players who make one move each.
- The goal is to find a *subgame perfect Nash equilibrium*, i.e., the move by each player that ensures that player's best outcome.

Recourse Game

- A cooperative game in which play alternates between cooperating players and chance players.
- The goal is to find a *subgame perfect Markov equilibrium*, i.e., the move that ensures the best outcome in a probabilistic sense.

Multilevel and Multistage Optimization

- A standard mathematical program models a (set of) decision(s) to be made *simultaneously* by a *single* decision-maker (i.e., with a *single* objective).
- Decision problems arising in sequential games and other real-world applications involve
  - multiple, independent decision-makers (DMs),
  - sequential/multi-stage decision processes, and/or
  - multiple, possibly conflicting objectives.
- Modeling frameworks
  - Multiobjective Programming $\Leftarrow$ multiple objectives, single DM
  - Mathematical Programming with Recourse $\Leftarrow$ multiple stages, single DM
  - Multilevel Programming $\Leftarrow$ multiple stages, multiple objectives, multiple DMs
- *Multilevel programming* generalizes standard mathematical programming by modeling hierarchical decision problems, such as finite extensive-form games.
- Such models arises in a remarkably wide array of applications.

Brief Overview of Practical Applications

- Hierarchical decision systems
  - Government agencies
  - Large corporations with multiple subsidiaries
  - Markets with a single "market-maker."
  - Decision problems with recourse
- Parties in direct conflict
  - Zero sum games
  - Interdiction problems
- Modeling "robustness": Chance player is external phenomena that cannot be controlled.
  - Weather
  - External market conditions
- Controlling optimized systems: One of the players is a system that is optimized by its nature.
  - Electrical networks
  - Biological systems

Outline

Setting: Two-Stage Mixed Integer Optimization

- We have the following general formulation:

2SMILP

$$z_{\text{2SMILP}} = \min_{x \in \mathcal{P}_1} \Psi(x) = \min_{x \in \mathcal{P}_1} \left\{ c^\top x + \Xi(x) \right\}, \tag{2SMILP}$$

where

$$\mathcal{P}_1 = \left\{ x \in X \mid A^1 x = b^1 \right\}$$

is the *first-stage feasible region* with $X = \mathbb{Z}_+^{r_1} \times \mathbb{R}_+^{n_1 - r_1}$, $A^1 \in \mathbb{Q}^{m_1 \times n_1}$, and $b^1 \in \mathbb{R}^{m_1}$.

- Ξ is a “risk function” that represents the impact of future uncertainty.
- We’ll refer to Ξ as the *second-stage risk function*.
- The uncertainty can arise either due to stochasticity or due to the fact that Ξ represents the reaction of a competitor.

Special Case I: Recourse Problems

- Recourse problems are a special case in which the risk function has a certain simple form.
- For example, the canonical form of Ξ employed in the case of two-stage stochastic integer programming is

Stochastic Risk Function

$$\Xi(x) = \mathbb{E}_{\omega \in \Omega}\left[\phi(h_\omega - T_\omega x)\right],$$

where ω is a random variable from a probability space $(\Omega, \mathscr{F}, \mathcal{P})$.

- For each $\omega \in \Omega$, $T_\omega \in \mathbb{Q}^{m_2 \times n_1}$ and $h_\omega \in \mathbb{Q}^{m_2}$ is the realization of the input to the second-stage problem for scenario ω.
- ϕ is the value function of the recourse MILP, to be defined next.

Related Work on Stochastic Programming with Recourse

	First Stage			Second Stage			Stochasticity			
	$\mathbb{R}$	$\mathbb{Z}$	$\mathbb{B}$	$\mathbb{R}$	$\mathbb{Z}$	$\mathbb{B}$	$\mathbf{W}$	$\mathbf{T}$	$\mathbf{h}$	$\mathbf{q}$
Laporte and Louveaux [1993]			$\ast$	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$	
Carøe and Tind [1997]	$\ast$		$\ast$	$\ast$		$\ast$	$\ast$	$\ast$	$\ast$	$\ast$
Carøe and Tind [1998]	$\ast$	$\ast$	$\ast$		$\ast$	$\ast$		$\ast$	$\ast$	
Carøe and Schultz [1998]	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$		$\ast$	$\ast$	$\ast$
Schultz et al. [1998]	$\ast$				$\ast$	$\ast$			$\ast$	
Sherali and Fraticelli [2002]			$\ast$	$\ast$		$\ast$	$\ast$	$\ast$	$\ast$	$\ast$
Ahmed et al. [2004]	$\ast$	$\ast$	$\ast$		$\ast$	$\ast$	$\ast$		$\ast$	$\ast$
Sen and Higle [2005]			$\ast$	$\ast$		$\ast$		$\ast$	$\ast$	
Sen and Sherali [2006]			$\ast$	$\ast$	$\ast$	$\ast$		$\ast$	$\ast$	
Sherali and Zhu [2006]	$\ast$		$\ast$	$\ast$		$\ast$	$\ast$	$\ast$	$\ast$	
Kong et al. [2006]		$\ast$	$\ast$		$\ast$	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$
Sherali and Smith [2009]			$\ast$	$\ast$		$\ast$	$\ast$	$\ast$	$\ast$	$\ast$
Yuan and Sen [2009]			$\ast$	$\ast$		$\ast$		$\ast$	$\ast$	$\ast$
Ntaimo [2010]			$\ast$	$\ast$		$\ast$	$\ast$			$\ast$
Gade et al. [2012]			$\ast$		$\ast$	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$
Trapp et al. [2013]		$\ast$	$\ast$		$\ast$	$\ast$			$\ast$	
Hassanzadeh and Ralphs [2014a]	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$	$\ast$		$\ast$	$\ast$	

The Second-Stage Value Function

- The structure of the objective function Ψ depends primarily on the structure of the *value function*

Second-stage Value (Recourse) Function

$$\phi(\beta) = \min_{y \in \mathcal{P}_2(\beta)} q^\top y \tag{2S-VF}$$

where

$$\mathcal{P}_2(\beta) = \{y \in Y \mid G^2 y = \beta\} \tag{1}$$

is the *second-stage feasible region* with respect to a given right-hand side β, $Y = \mathbb{Z}_+^{r_2} \times \mathbb{R}_+^{n_2 - r_2}$ and $G^2 \in \mathbb{Q}^{m_2 \times n_2}$.

- The second-stage problem is parameterized on the unknown value β of the right-hand side.
- This value is determined jointly by the realized value of ω and the values of the first-stage decision variables.

Related Work on Value Function

Duality

- Johnson [1973, 1974, 1979]
- Jeroslow [1979]
- Wolsey [1981]
- Güzelsoy and Ralphs [2007], Güzelsoy [2009]

Structure and Construction

- Blair and Jeroslow [1977, 1982], Blair [1995]
- Kong et al. [2006]
- Hassanzadeh and Ralphs [2014b]

Sensitivity and Warm Starting

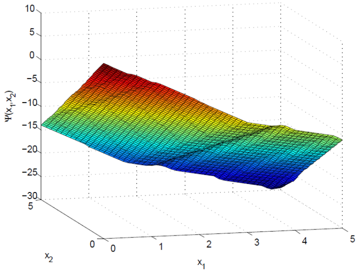
- Ralphs and Güzelsoy [2005, 2006], Güzelsoy [2009]
- Gamrath et al. [2015]

Example

Example 1

$$\begin{aligned} \min \Psi(x_1, x_2) = \min \ & -3x_1 - 4x_2 + \mathbb{E}[\phi(\omega - 2x_1 - 0.5x_2)] \\ s.t.\ & x_1 \leq 5, x_2 \leq 5 \\ & x_1, x_2 \in \mathbb{R}_+, \end{aligned} \tag{Ex.SMP}$$

and $\omega \in \{6, 12\}$ *with a uniform probability distribution.*



Special Case II: Bilevel (Integer) Linear Optimization

In the case of general bilevel optimization, we have

Bilevel Risk Function

$$\Xi(x) = \min_{y \in \mathcal{P}_2(b^2 - A^2 x) \cap Y} \left\{ d^1 y \mid d^2 y = \phi(b^2 - A^2 x) \right\}$$

where $A^2 \in \mathbb{Q}^{m_2 \times n_1}$, and $b^2 \in \mathbb{R}^{m_2}$, $\mathcal{P}_2(\beta) = \left\{ y \in \mathbb{R}_+ \mid G^2 y \geq \beta \right\}$, and $Y = \mathbb{Z}^{p_2} \times \mathbb{R}^{n_2 - p_2}$.

Alternatively, the more familiar and equivalent form of the problem is

Mixed Integer Bilevel Linear Optimization Problem (MIBLP)

$$\min \left\{ cx + d^1 y \mid x \in \mathcal{P}_1 \cap X, y \in \operatorname{argmin}\{ d^2 y \mid y \in \mathcal{P}_2(b^2 - A^2 x) \cap Y \} \right\} \tag{MIBLP}$$

Note that this is well-defined to be the **optimistic case**.

Related Work On Bilevel Optimization

General Nonconvex

- Mitsos [2010]
- Kleniati and Adjiman [2014a,b]

Discrete Linear

- Moore and Bard [1990]
- DeNegre [2011], DeNegre and Ralphs [2009], DeNegre et al. [2016]
- Xu [2012]
- Caramia and Mari [2013]
- Caprara et al. [2014a]
- Fischetti et al. [2016]
- Hemmati and Smith [2016], Lozano and Smith [2016]

Other Special Cases

- Pure integer.
- Positive constraint matrix at second stage.
- Binary variables at the first and/or second stage.
- Zero sum and interdiction problems.

Mixed Integer Interdiction

$$\max_{x \in \mathcal{P}_1 \cap X} \min_{y \in \mathcal{P}_2(x) \cap Y} dy \tag{MIPINT}$$

where

$$\begin{aligned} \mathcal{P}_1 &= \left\{ x \in X \mid A^1 x \leq b^1 \right\} & X &= \mathbb{B}^n \\ \mathcal{P}_2(x) &= \left\{ y \in Y \mid G^2 y \geq b^2, y \leq u(e - x) \right\} & Y &= \mathbb{Z}^p \times \mathbb{R}^{n-p} \end{aligned}$$

- The case where follower's problem has network structure is called the *network interdiction problem* and has been well-studied.
- The model above allows for second-stage systems described by general MILPs.

Complexity

- When the second-stage problem is NP-hard, the two-stage problem is not even in NP.
- In fact, for even simple cases (knapsack interdiction), these problems are hard for the second level of the polynomial hierarchy.
- Formally, the decision version of the problem is complete for Σ_2^p[1].
- See Lodi et al. [2014], Caprara et al. [2014b].
- Details will be in a paper Leo and I are going to write[2].

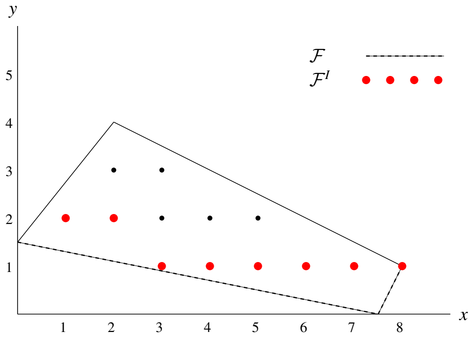
[1] Also known as "Chuck Norris-hard."

[2] Leo is going to write it and I'm going to put my name on it

Outline

Value Function Reformulation

More generally, we can reformulate (MIBLP) as



$$
\begin{aligned}
\min \quad & c^1x + d^1y \\
\text{subject to} \quad & A^1x \leq b^1 \\
& G^2y \geq b^2 - A^2x \\
& d^2y \leq \phi(b^2 - A^2x) \\
& x \in X, y \in Y,
\end{aligned}
$$

where ϕ is the value function of the second-stage problem.

- This is, in principle, a standard mathematical program.
- Note that the second-stage variables need to appear in the formulation in order to enforce feasibility.

Value Function Reformulation (Recourse)

An important special case is when $d^1 = d^2$. We can reformulate (MIBLP) as

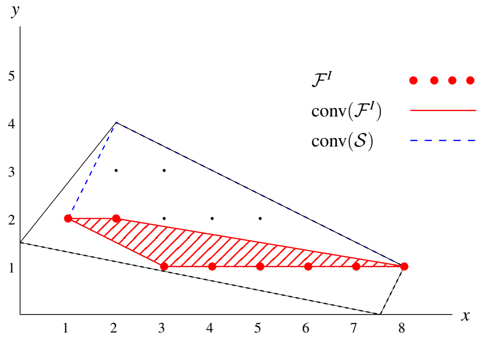
$$\begin{aligned} \min \quad & c^1 x + w \\ \text{subject to} \quad & A^1 x \le b^1 \\ & w \ge \phi(b^2 - A^2 x) \\ & x \in X \end{aligned}$$

where ϕ is again the value function of the second-stage problem.

- In these special cases, we can project out the second-stage variables, as in a traditional Benders algorithm.
- This leads to a generalized Benders algorithm obtained by constructing approximations of ϕ dynamically.
- Non-linear “cuts” arising from lower-bounding functions are added in each step to improve the approximations.

Polyhedral Reformulation

Convexification considers the following conceptual reformulation.



$$\begin{aligned} \min \quad & c^1 x + d^1 y \\ \text{s.t.} \quad & (x, y) \in \text{conv}(\mathcal{F}^I) \end{aligned}$$

where $\mathcal{F}^I = \{(x, y) \mid x \in \mathcal{P}_1 \cap X, y \in \text{argmin}\{d^2 y \mid y \in \mathcal{P}_2(x) \cap Y\}\}$

- To get bounds, we'll optimize over a relaxed feasible region.
- We'll iteratively approximate the true feasible region with linear inequalities.

Roadmap for the Talk

- We'll discuss two main classes of algorithms

Dual

- Generalized Benders approach
- Approximate the value function from **below**.
- "Benders cuts" are (non-linear, non-convex) "dual functions".
- Can be combined with branching to get "local convexity".

Primal

- Generalized branch-and-cut approach
- Approximate the value function from **above**
- With linear "optimality cuts", we need to branch to achieve convergence in the general case.

- Naturally, we can also have hybrids.
- Any convergent algorithm for bilevel optimization must somehow construct an approximation of the value function, usually by intelligent "sampling."

Outline

MILP Value Function

- In this section, we explore the structure of the value function associated a mixed integer linear optimization problem

$$z_{IP} = \min_{x \in \mathcal{S}} c^\top x, \tag{MILP}$$

where, $c \in \mathbb{R}^n$, $S = \{x \in \mathbb{Z}^r_+ \times \mathbb{R}^{n-r}_+ \mid Ax = b\}$ with $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{R}^m$.

- The *value function* associated with the base instance (MILP) is

MILP Value Function

$$\phi(\beta) = \min_{x \in \mathcal{S}(\beta)} c^\top x \tag{MILP-VF}$$

for $\beta \in \mathbb{R}^m$, where $\mathcal{S}(\beta) = \{x \in \mathbb{Z}^r_+ \times \mathbb{R}^{n-r}_+ \mid \; Ax = \beta\}$.

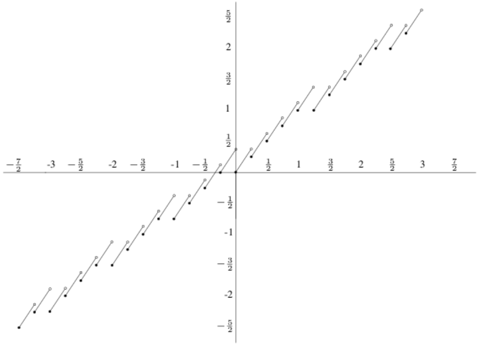
- We let $\phi(\beta) = \infty$ if $\beta \in \Omega = \{\beta \in \mathbb{R}^m \mid \mathcal{S}(\beta) = \emptyset\}$.

Properties of the MILP Value Function

The MILP value function is non-convex, discontinuous, and piecewise polyhedral.

Example 2

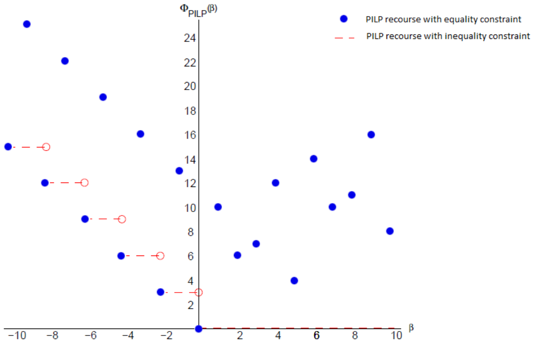
$$
\begin{aligned}
\phi(\beta) = \min\ & x_1 - \frac{3}{4}x_2 + \frac{3}{4}x_3 \\
s.t.\ & \frac{5}{4}x_1 - x_2 + \frac{1}{2}x_3 = \beta \\
& x_1, x_2 \in \mathbb{Z}_+,\ x_3 \in \mathbb{R}_+
\end{aligned}
\tag{Ex2.MILP}
$$



Example: MILP Value Function (Pure Integer)

Example 3

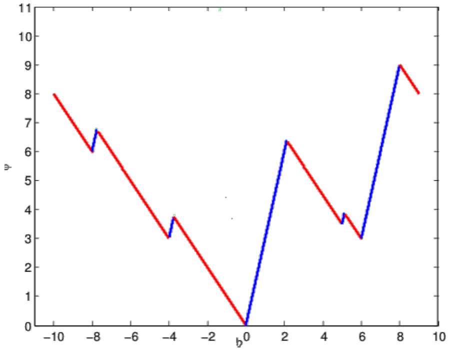
$$\phi(\beta) = \min 3x_1 + \frac{7}{2}x_2 + 3x_3 + 6x_4 + 7x_5 + 5x_6$$
$$s.t.\ 6x_1 + 5x_2 - 4x_3 + 2x_4 - 7x_5 + x_6 = \beta$$
$$x_1, x_2, x_3, x_4, x_5, x_6 \in \mathbb{Z}_+$$



Another Example

Example 4

$$\phi(\beta) = \min 3x_1 + \frac{7}{2}x_2 + 3x_3 + 6x_4 + 7x_5 + 5x_6$$
$$s.t.\ 6x_1 + 5x_2 - 4x_3 + 2x_4 - 7x_5 + x_6 = \beta$$
$$x_1, x_2, x_3 \in \mathbb{Z}_+,\ x_4, x_5, x_6 \in \mathbb{R}_+$$



Continuous and Integer Restriction of an MILP

Consider the general form of the second-stage value function

$$\begin{aligned} \phi(\beta) = \min\, & c_I^\top x_I + c_C^\top x_C \\ \text{s.t. } & A_I x_I + A_C x_C = \beta, \\ & y \in \mathbb{Z}_+^{r_2} \times \mathbb{R}_+^{n_2 - r_2} \end{aligned} \tag{MILP}$$

The structure is inherited from that of the *continuous restriction*:

$$\begin{aligned} \phi_C(\beta) = \min\, & c_C^\top x_C \\ \text{s.t. } & A_C x_C = \beta, \\ & x_C \in \mathbb{R}_+^{n_2 - r_2} \end{aligned} \tag{CR}$$

for $C = \{p_2 + 1, \ldots, n_2\}$ and the similarly defined *integer restriction*:

$$\begin{aligned} \phi_I(\beta) = \min\, & c_I^\top x_I \\ \text{s.t. } & A_I x_I = \beta \\ & x_I \in \mathbb{Z}_+^{r_2} \end{aligned} \tag{IR}$$

for $I = \{p_2 + 1, \ldots, n_2\}$.

Discrete Representation of the Value Function

For $\beta \in \mathbb{R}^{m_2}$, we have that

$$\begin{aligned} \phi(\beta) = \min\ & c_I^\top x_I + \phi_C(\beta - A_I x_I) \\ \text{s.t. } & x_I \in \mathbb{Z}_+^{r_2} \end{aligned} \tag{2}$$

- From this we see that the value function is comprised of the minimum of a set of translations of ϕ_C.
- The set of shifts, along with ϕ_C describe the value function exactly.
- For $\hat{x}_I \in \mathbb{Z}_+^{r_2}$, let

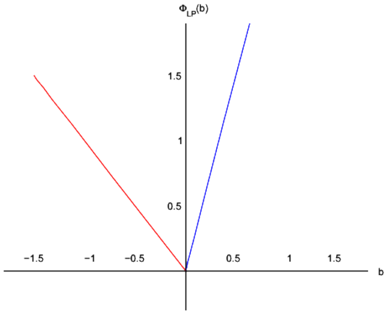
$$\phi_C(\beta, \hat{x}_I) = c_I^\top \hat{x}_I + \phi_C(\beta - A_I \hat{x}_I)\ \forall \beta \in \mathbb{R}^{m_2}. \tag{3}$$

- Then we have that $\phi(\beta) = \min_{x_I \in \mathbb{Z}_+^{r_2}} \phi_C(\beta, \hat{x}_I)$.

Value Function of the Continuous Restriction

Example 5

$$\begin{aligned}\phi_C(\beta) = \min\ & 6y_1 + 7y_2 + 5y_3 \\ s.t.\ & 2y_1 - 7y_2 + y_3 = \beta \\ & y_1, y_2, y_3 \in \mathbb{R}_+\end{aligned}$$



Related Results

- From the basic structure outlined, we can derive many other useful results.

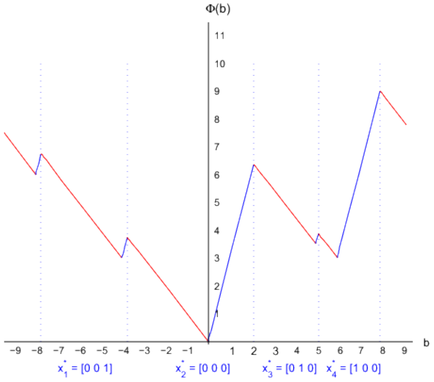
Proposition 1 *[Hassanzadeh and Ralphs, 2014b] The gradient of ϕ on a neighborhood of a differentiable point is a unique optimal dual feasible solution to* (CR).

Proposition 2 *[Hassanzadeh and Ralphs, 2014b] Consider $\mathcal{N} \subseteq \mathbb{R}^m$ over which ϕ is differentiable. Then, there exist an integral part of the solution $x_I^* \in \mathbb{Z}^r$ and $E \in \mathcal{E}$ such that $\phi(b) = c_I^\top x_I^* + \nu_E^\top (b - A_I x_I^*)$ for all $b \in \mathcal{N}$.*

- This last result can be extended to subset of the domain over which ϕ is convex.
- Over such a region, ϕ coincides with the value function of a translation of the continuous restriction.
- Putting all of together, we get practical finite representation...

Points of Strict Local Convexity (Finite Representation)

Example 6



Theorem 1 *[Hassanzadeh and Ralphs, 2014b]*
Under the assumption that $\{\beta \in \mathbb{R}^{m_2} \mid \phi_I(\beta) < \infty\}$ *is finite, there exists a finite set* $\mathcal{S} \subseteq Y$ *such that*

$$\phi(\beta) = \min_{x_I \in \mathcal{S}} \{c_I^\top x_I + \phi_C(\beta - A_I x_I)\}. \tag{4}$$

Outline

Approximating the Value Function

- In general, it is difficult to construct the value function explicitly.
- We therefore propose to approximate the value function by either primal (upper) or dual (lower) bounding functions.

Dual bounds

Derived by considering the value function of *relaxations* of the original problem or by constructing *dual functions* $\Rightarrow$ Relax constraints.

Primal bounds

Derived by considering the value function of *restrictions* of the original problem $\Rightarrow$ Fix variables.

Dual Bounding Functions

- A *dual function* $F : \mathbb{R}^m \to \mathbb{R}$ is one that satisfies $F(\beta) \leq \phi(\beta)$ for all $\beta \in \mathbb{R}^m$.
- These can be seen as solutions to the following general dual.

$$\max \left\{F(b) : F(\beta) \leq \phi(\beta),\ \beta \in \mathbb{R}^m, F \in \Upsilon^m\right\} \tag{D}$$

where $\Upsilon^m \subseteq \{f \mid f : \mathbb{R}^m \to \mathbb{R}\}$.

- These dual functions are a generalization of the LP dual and play the role of the "cuts" in the Benders algorithm.

Dual Functions from Branch-and-Bound [Wolsey, 1981]

- Let T be the set of terminating nodes of the tree. Then in node $t \in T$, we solve:

$$\begin{aligned} \phi^t(\beta) = \min\ & c^\top x \\ \text{s.t. } & Ax = \beta, \\ & l^t \le x \le u^t, x \ge 0 \end{aligned} \tag{5}$$

- The dual of this LP at node t is:

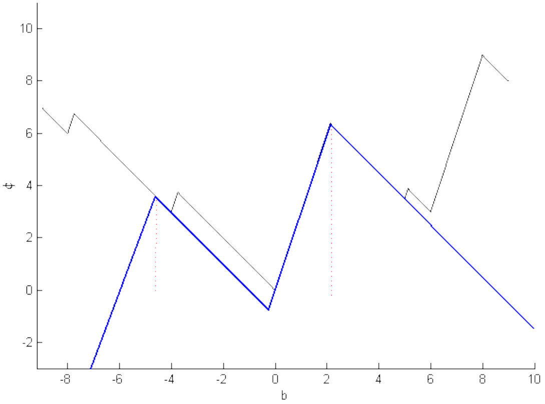
$$\begin{aligned} \phi^t(\beta) = \max\ & \pi^t \beta + \underline{\pi}^t l^t + \bar{\pi}^t u^t \\ \text{s.t. } & \pi^t A + \underline{\pi}^t + \bar{\pi}^t \le c^\top \\ & \underline{\pi} \ge 0, \bar{\pi} \le 0 \end{aligned} \tag{6}$$

- We obtain the following strong dual function:

$$\min_{t \in T} \{ \hat{\pi}^t \beta + \hat{\underline{\pi}}^t l^t + \hat{\bar{\pi}}^t u^t \}, \tag{7}$$

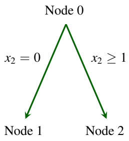
  where $(\hat{\pi}^t, \hat{\underline{\pi}}^t, \hat{\bar{\pi}}^t)$ is an optimal solution to the dual (6).
- This can be further strengthened by including internal nodes.

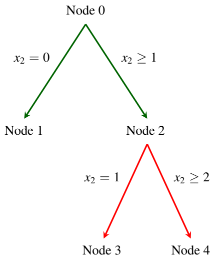
Dual Functions from Branch-and-Bound

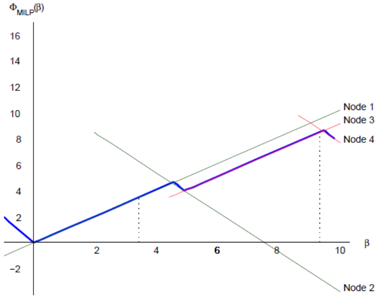


Iterative Refinement

- The tree obtained from evaluating $\phi(\beta)$ yields a dual function strong at β.
- By solving for other right-hand sides, we obtain additional dual functions that can be aggregated.
- These additional solves can be done **within the same tree**, eventually yielding a single tree representing the entire function.







Aside: Warm Starting and Sensitivity Analysis

- A procedure similar to iterative refinement be used to warm start the solution process for solving MILPs.
- We have been experimenting with the use of such warm-starting procedures to improve the solution times for bilevel feasibility checking.
- The dual functions obtained by solving one instance can be used in place of the usual linear dual for things like improving variable bounds.
- These dual functions can also be used for sensitivity analysis.
- This is the topic of another talk.

Tree Representation of the Value Function

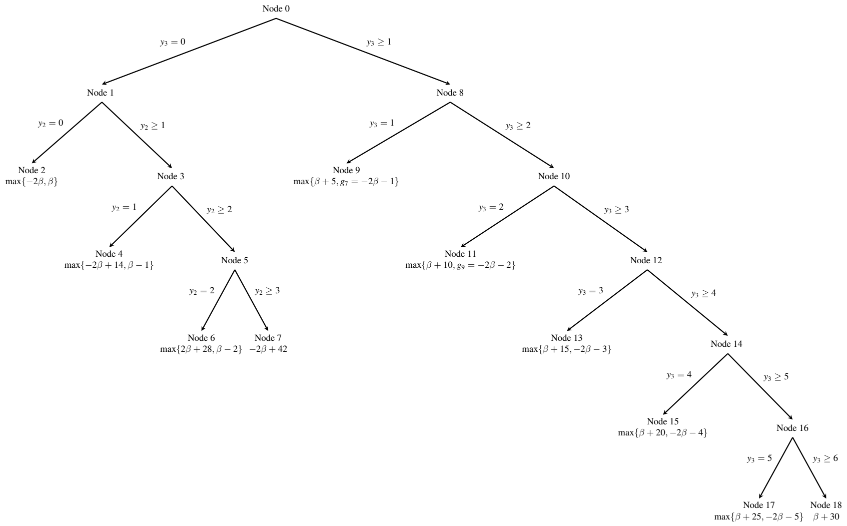
- Continuing the process, we eventually generate the entire value function.
- Consider the strengthened dual

$$\underline{\phi}^*(\beta) = \min_{t \in T} q_{I_t}^\top y_{I_t}^t + \phi_{N \setminus I_t}^t(\beta - W_{I_t} y_{I_t}^t), \tag{8}$$

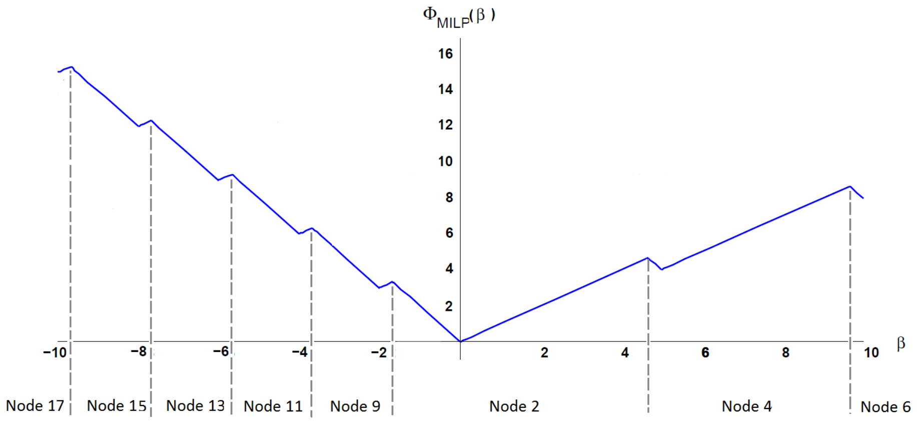
- I_t is the set of indices of fixed variables, $y_{I_t}^t$ are the values of the corresponding variables in node t.
- $\phi_{N \setminus I_t}^t$ is the value function of the linear program at node t including only the unfixed variables.

Theorem 2 *[Hassanzadeh and Ralphs, 2014a] Under the assumption that* $\{\beta \in \mathbb{R}^{m_2} \mid \phi_I(\beta) < \infty\}$ *is finite, there exists a branch-and-bound tree with respect to which* $\underline{\phi}^* = \phi$.

Example of Value Function Tree



Correspondence of Nodes and Local Stability Regions



Primal Bounding Functions

- Just like the dual (lower) bounding functions, we are interested in a function that is a valid primal (upper) bound for the value function.

Theorem 3 *Let x^* be an optimal solution to the primal problem with right-hand side b. Then $\phi_C(\beta, \hat{x}_I)$ is a strong primal bounding function at b.*

- By repeatedly evaluating $\phi_I(\beta)$, we can obtain upper approximations (and eventually the full value function).

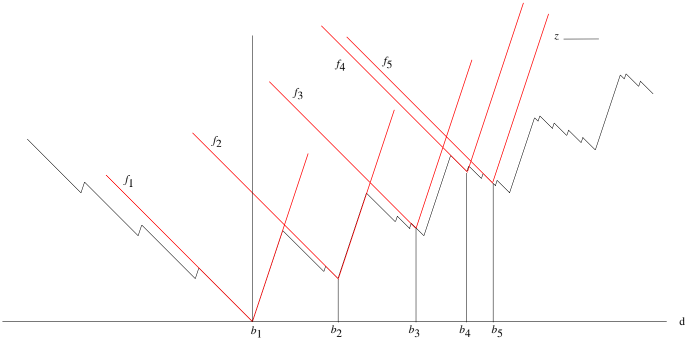


Figure: Primal bounding functions obtained at right-hand sides $b_i, i = 1, \ldots, 5$.

"Cutting" Algorithm for Generating the Value Function

Algorithm

```
Initialize: Let $\bar{\phi}(b) = \infty$ for all $b \in B$, $\Gamma^0 = \infty$, $x_I^0 = 0$, $S^0 = \{x_I^0\}$, and $k = 0$.
while $\Gamma^k > 0$ do:
    • Let $\bar{\phi}(\beta) \leftarrow \min\{\bar{\phi}(\beta), \bar{\phi}(\beta; x_I^k)\}$ for all $\beta \in \mathbb{R}^m$.
    • $k \leftarrow k + 1$.
    • Solve
          $$\Gamma^k = \max_{\beta \in \mathbb{R}^m} \bar{\phi}(\beta) - c_I^\top x_I \quad \text{s.t. } A_I x_I = b, \quad x_I \in \mathbb{Z}^r_+. \qquad \text{(SP)}$$
      to obtain $x_I^k$.
    • Set $S^k \leftarrow S^{k-1} \cup \{x^k\}$
end while
return $\phi(b) = \bar{\phi}(b)$ for all $b \in B$.
```

Formulating (SP)

Surprisingly, the “cut generation” problem (SP) can be formulated easily as an MINLP.

$$\begin{aligned} \Gamma^k = \; & \max \theta \\ \text{s.t. } & \theta + c_I^\top x_I \leq c_I^\top x_I^i + (A_I x_I - A_I x_I^i)^\top \nu^i \quad i = 1, \ldots, k-1 \\ & A_C^\top \nu^i \leq c_C \quad i = 1, \ldots, k-1 \\ & \nu^i \in \mathbb{R}^m \quad i = 1, \ldots, k-1 \\ & x_I \in \mathbb{Z}_+^r . \end{aligned} \tag{9}$$

Sample Computational Results

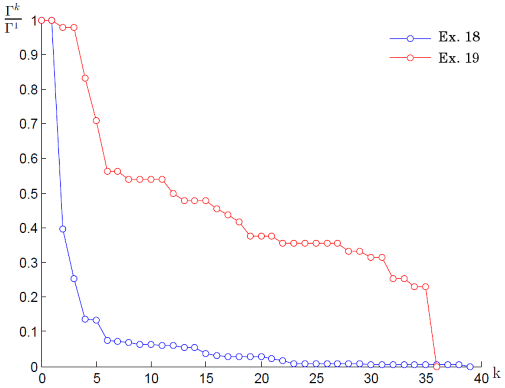


Figure: Normalized approximation gap vs. iteration number.

```
http://github.com/tkralphs/ValueFunction
```

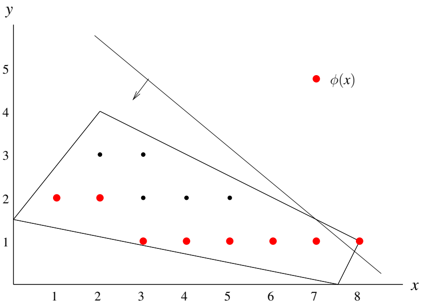
Outline

Benders' Principle

$$\begin{aligned} z_{\text{IP}} &= \min_{(x,y)\in\mathbb{Z}^n} \left\{ c'x + c''y \;\middle|\; A'x + A''y \geq b \right\} \\ &= \min_{x\in\mathbb{R}^{n'}} \left\{ c'x + \phi(b - A'x) \right\}, \end{aligned}$$

where

$$\begin{aligned} \phi(d) = \quad & \min c''y \\ & \text{s.t. } A''y \geq d \\ & y \in \mathbb{Z}^{n''} \end{aligned}$$



Basic Strategy:

- ϕ is the value function of an integer program.
- We iteratively generate an approximation of ϕ by constructing dual functions.

Generalized Benders in a Nutshell [Hassanzadeh and Ralphs, 2014a]

Benders' Master Problem

$$\begin{aligned} \min \quad & c'x + w \\ \text{subject to} \quad & A'x \leq b' \\ & w \geq \underline{\phi}(b' - A'x) \\ & x \in X \end{aligned}$$

- $\underline{\phi}$ is a lower approximation of the value function of the second stage problem.
- We have projected out the second-stage variables.
- if $Y = \mathbb{R}^{n''}$, ϕ is a piece-wise linear convex function.
- The classical Benders' algorithm approximates ϕ as the maximum of linear dual functions, which are the dual solutions to a linear program.
- Our method follows the same outline, but our "dual functions" will need to be non-linear.
- The remainder of the talk is about our computational framework for constructing and representing these functions.

Outline

Implementation Overview

Basic Scheme

1. Solve master problem to obtain new first stage solution and lower bound.
2. Solve scenario subproblems to update value function approximation and obtain new upper bound.
3. Terminate when upper bound equals lower bound.

- As in the classical algorithm, we alternate between solving the master problem and subproblems that update the current approximation.
- The approximation may come from a single tree or a set of trees (more shortly).
- We require a solver capable of exporting the dual function resulting from the solve process.
- Ideally, the solver should also be capable of iterative refinement and warm-starting, though this is not necessary.
- The SYMPHONY MILP solver has this capability.

Quick Example

Consider

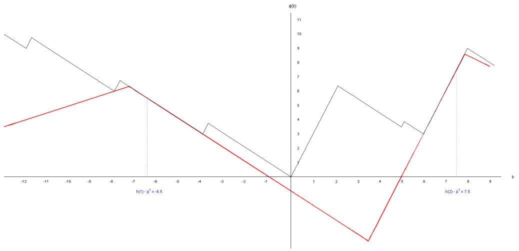
$$\begin{aligned} \min \Psi(x) = \min\ & -3x_1 - 4x_2 + \sum_{\omega=1}^{2} 0.5\phi(h_\omega - T_\omega x) \\ \text{s.t. } & x_1 + x_2 \le 5 \\ & x \in \mathbb{Z}_+ \end{aligned} \tag{10}$$

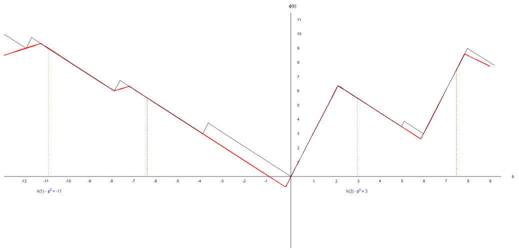
where

$$\begin{aligned} \phi(\beta) = \min\ & 3y_1 + \frac{7}{2}y_2 + 3y_3 + 6y_4 + 7y_5 \\ \text{s.t. } & 6y_1 + 5y_2 - 4y_3 + 2y_4 - 7y_5 = \beta \\ & y_1, y_2, y_3 \in \mathbb{Z}_+,\ y_4, y_5 \in \mathbb{R}_+ \end{aligned} \tag{11}$$

with $h_\omega = [-4 \ \ 10]$ and $T_\omega = [-2 \ \ \frac{1}{2}]^\top$ (technology matrix is constant)).

Example





Outline

SSLP Instances

We apply the algorithm to the SSLP test instances from SIPLIB. The instances have the following properties. The last column shows the deterministic equivalent solution time in seconds.

	DEP			2nd Stage				
Instance	cons	bin	int	cons	bin	int	Time (s)	% Gap
sslp-5-25(25)	751	3130	125	30	130	5	3.17	0
sslp-5-25(50)	1501	6255	250	30	130	5	4.22	0
sslp-5-25(100)	3001	12505	500	30	130	5	14.34	0
sslp-10-50(50)	3001	25010	500	60	510	10	3600+	70
sslp-10-50(100)	6001	50010	1000	60	510	10	3600+	15
sslp-15-45(5)	301	3390	75	60	690	15	3600+	1
sslp-15-45(10)	601	6765	150	60	690	15	1088.69	0

Table: The deterministic equivalent of SSLP instances

where “DEP” and “2nd Stage” correspond to the deterministic equivalent and the second stage problems and “cons”, “bins” and “int” respectively represent the number of constraints, binary variables and general integer variables in the corresponding problem.

SSLP Instances

Instance	Iteration	Size	Time (s)	%Gap
sslp-5-25(25)	16	(2493, 3639)	182.58	0
sslp-5-25(50)	18	(789, 1821)	12.16	0
sslp-5-25(100)	18	(1322, 3410)	27.91	0
sslp-10-50(50)	8	(40K, 71K)	-	23
sslp-10-50(100)	8	(74K, 125K)	-	9
sslp-15-45(5)	7	(29K, 56K)	-	99
sslp-15-45(10)	26	(17K, 29K)	-	52

Table: Generalized Benders' algorithm applied to SSLP instances

- The results are generated using the MILP solver SYMPHONY version WS revision 2522 and CPLEX 12.5.
- The tests were performed on a 16-core Linux box with 800 MHz AMD processors and 31 GB RAM compiled with g++.
- SSLP 10 and 15 runs were in parallel using 16 cores, other runs sequential.
- The "Gap%" column refers to the relative gap between the upper bound and lower bound of the Generalized Benders' algorithm.

Analysis

- Getting this all to work well is a PITA!

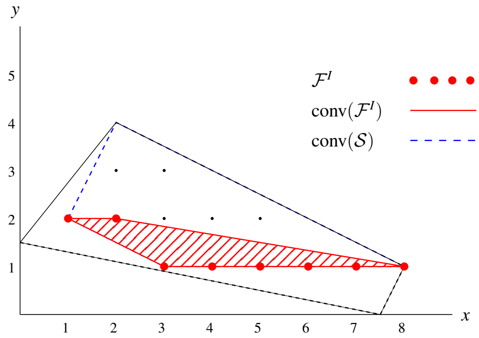
Figure: Jack-in-the-Box Chicken Fajita Pita

- It appears possible to generalize the master problem to work for more general bilevel problems, but there are some potential stumbling blocks.

Outline

Basic Convexification Strategy (cont'd)

Convexification considers the following conceptual reformulation.



$$\begin{aligned} \min \quad & c^1 x + d^1 y \\ \text{s.t.} \quad & (x,y) \in \mathrm{conv}(\mathcal{F}^I) \end{aligned}$$

where $\mathcal{F}^I = \{(x,y) \mid x \in \mathcal{P}_1 \cap X, y \in \mathrm{argmin}\{d^2 y \mid y \in \mathcal{P}_2(x) \cap Y\}\}$

- To get bounds, we'll optimize over a relaxed feasible region.
- We'll iteratively approximate the true feasible region with linear inequalities.
- This turns out to be essentially the same as bounding the value function from above with linear function.

A Branch-and-Cut Algorithm

- Putting all of this together, we propose a branch-and-bound approach.

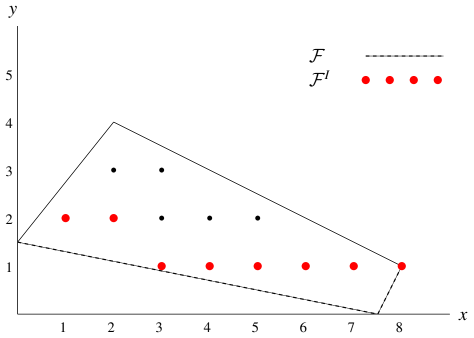
Components

- Bounding
- Branching
- Feasibility checking
- Search strategies
- Preprocessing methods
- Primal heuristics

- In the remainder of the talk, we address development of these components, focusing mainly on bounding.

Dual Bounds

Dual bounds for the MIBLP can be obtained by relaxing the value function constraint.



$$
\begin{aligned}
\min \quad & c^1 x + d^1 y \\
\text{subject to} \quad & A^1 x \leq b^1 \\
& G^2 y \geq b^2 - a^2 x \\
& H^i x + H^2 y \leq h \\
& x \in X, y \in Y,
\end{aligned}
$$

- Note that in practice, we may further relax integrality conditions.
- The additional inequalities are valid inequalities that serve to approximate the value function.
- The algorithm is very similar to branch-and-cut for solving traditional mathematical programs.

Bilevel Feasibility Check

- Let $(\hat{x}, \hat{y}) \in X \times Y$ be a solution to the dual bounding relaxation problem.
- We fix $x = \hat{x}$ and solve the second-stage problem

$$\min_{y \in \mathcal{P}_2(\hat{x}) \cap X} d^2 y \tag{12}$$

  with the fixed first-stage solution $\hat{x}$.
- Let y^* be the solution to (12).
  - $(\hat{x}, y^*)$ is bilevel feasible $\Rightarrow c^1\hat{x} + d^1 y^*$ is a valid primal bound on the optimal value of the original MIBLP
  - Either
    1. $d^2\hat{y} = d^2 y^* \Rightarrow (\hat{x}, \hat{y})$ is bilevel feasible.
    2. $d^2\hat{y} > d^2 y^* \Rightarrow (\hat{x}, \hat{y})$ is bilevel infeasible.
- What do we do in the case of bilevel infeasibility?
  - Generate a valid inequality violated by $(\hat{x}, \hat{y})$ (improve our approximation of the value function).
  - Branch on a disjunction violated by $(\hat{x}, \hat{y})$.

Gomory-like Cuts (Pure Integer Case)

- We can generate cuts to separate solutions proven bilevel infeasible using a procedure similar to that for generating *Gomory cuts*.
- Let

$$A := \begin{bmatrix} A^1 \\ A^2 \end{bmatrix} \in \mathbb{Z}^{(m_1+m_2)\times n_1}, G := \begin{bmatrix} 0 \\ G^2 \end{bmatrix} \in \mathbb{Z}^{m_2\times n_2}, b := \begin{bmatrix} b^1 \\ b^2 \end{bmatrix} \in \mathbb{Z}^{m_1+m_2}.$$

A basic feasible solution $(\hat{x}, \hat{y}) \in \Omega^I$ to the dual bounding problem is the *unique* solution to

$$a_i'x + g_i'y = b_i, \quad i \in I$$

where I is the set of active constraints at $(\hat{x}, \hat{y})$.

- This implies that

$$\left\{(x,y) \in \Omega^I \mid \sum_{i\in I} a_i'x + g_i'y = \sum_{i\in I} b_i\right\} = \left\{(\hat{x}, \hat{y})\right\}$$

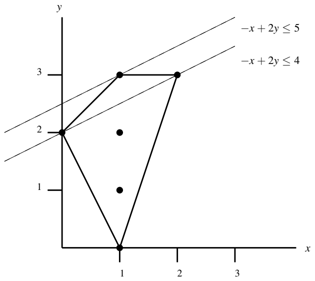
and $\sum_{i\in I} a_i'x + g_i'y \leq \sum_{i\in I} b_i$ is valid for Ω.

Gomory-like Cuts (cont'd.)

A Valid Inequality

$\sum_{i\in I} a_i' x + g_i' y \leq \sum_{i\in I} b_i - 1$ for all $(x,y) \in \Omega^I \setminus \{(\hat{x},\hat{y})\}$.

$$\max_x \min_y \{y \mid -x+y \leq 2, -2x-y \leq -2, 3x-y \leq 3, y \leq 3, x, y \in \mathbb{Z}_+\}.$$



This yields a finite algorithm in the pure integer case. We can generalize these cuts to the case where the constraint matrix is rational.

No Good Cuts

- In the binary case, we can directly enforce $x \neq \hat{x}$ by imposing the cut

$$\sum_{i:\hat{x}_i=0} x_i + \sum_{i:\hat{x}_i=1} (1 - x_i) \geq 1.$$

- This also cuts one solution at a time.

Optimality Cuts

- Stronger cuts can be obtained by exploiting the **bound information obtained during the feasibility check**.
- Implicitly, we will impose the constraint

$$d^2 y \leq \bar{\phi}(b^2 - A^2 x)$$

by adding a set of linear cuts (which may be locally or globally valid).

- In order to accomplish this, we need to do it in tandem with branching—impose cuts that are locally valid to overcome nonconvexity.
- After checking bilevel feasibility of $(x, y) \in (\mathcal{P}_1 \cap X) \times (\mathcal{P}_2(x) \cap Y)$, we know that

$$\hat{y} \in \mathcal{P}_2(x) \Rightarrow d^2 y \leq d^2 \hat{y}$$

- There are a number of ways to impose this logic.
  - Generate intersection cuts [Fischetti et al., 2016].
  - Impose the logic with integer variables [Mitsos, 2010]

A Branching Disjunction

- We then have the following valid disjunction [Xu, 2012].

$$\begin{array}{rcl}
(A^2 x \leq b^2 - G^2 \hat{y} & \text{AND} & d^\top y \leq d^\top \hat{y}) \text{ OR} \\
(A^2 x)_1 & > & b^2_1 - (G^2 \hat{y})_1 \text{ OR} \\
(A^2 x)_2 & > & b^2_2 - (G^2 \hat{y})_2 \text{ OR} \\
 & \vdots & \\
(A^2 x)_{m_2} & > & b^2_{m_2} - (G^2 \hat{y})_{m_2} \text{ OR}
\end{array}$$

- This can be imposed in practice by branching with appropriate choice of ϵ's to enforce strict inequality.

Special Case: $G^2 \in \mathbb{R}_+^{m_2 \times n_2}$

- When $G^2 \in \mathbb{R}_+^{m_2 \times n_2}$, we have the property that

$$x \geq \hat{x} \Rightarrow \mathcal{P}_2(x) \subseteq \mathcal{P}_2(\hat{x})$$

- This means that

$$x \geq \hat{x} \quad \Rightarrow \quad \phi(G^2 - A^2 x) \geq \phi(G^2 - A^2 \hat{x})$$

- If we additionally have binary upper level variables, then the previous disjunction can be simplified to

$$\sum_{i:\hat{x}_i=0} x_i = 0 \qquad \text{AND} \qquad d^2 y \leq d^2 \hat{y}$$

$$\text{OR}$$

$$\sum_{i:\hat{x}_i=0} x_i \quad \geq \quad 1$$

Special Case: Interdiction

- In the case of interdiction problems, we have

$$x_i = 1 \Rightarrow y_i = 0$$

- When $G^2 \in \mathbb{R}_+^{m_2 \times n_2}$, we also have that $G^2 \hat{y} \leq b^2 \Rightarrow G^2 y \leq b^2 \ \forall y \leq \hat{y}$.
- We can then derive a cut

$$d^2 y \leq \sum_{1 \leq i \leq n_2} d_i^2 \hat{y}_i (1 - x_i)$$

- We call these *Benders cuts* because if we add one such cut for each second-stage solution, we get the exact value function.
- This cut was mentioned (but not used) in Caprara et al. [2014a]'s work on knapsack interdiction.
- Anecdotally seems to have been used previously in some application-oriented papers.

More Generally

- More generally, what we would like to do is to impose the following constraint.

$$d^2 y \leq \bar{\phi}(b^2 - A^2 x)$$

where $\bar{\phi}$ is an *upper approximation* of the value function.

- A naive attempt would be to calculate a global bound α on the second-stage objective value and impose a cut $d^2 y \leq \alpha$.

Bilevel Bounding Problem

$$\begin{aligned} \alpha = \max \quad & d^2 y \\ \text{s.t.} \quad & A^1 x \leq b^1 \\ & y \in \operatorname{argmin}\{d^2 y : G^2 y \leq b^2 - A^2 x, y \in Y\} \\ & x \in X \end{aligned}$$

- This initial inequality can be lifted or we can generate locally valid versions at nodes in the tree.
- Note that we do not need the optimal value α^*—any upper bound will suffice.

Outline

Implementation: MibS

The *Mixed Integer Bilevel Solver* (MibS) implements the branch and bound framework described here using software available from the Computational Infrastructure for Operations Research (COIN-OR) repository.

COIN-OR Components Used

- The COIN High Performance Parallel Search (CHiPPS) framework to manage the global branch and bound.
- The SYMPHONY framework for checking bilevel feasibility..
- The COIN LP Solver (CLP) framework for solving the LPs arising in the branch and cut.
- The Cut Generation Library (CGL) for generating cutting planes within both SYMPHONY and MibS itself.
- The Open Solver Interface (OSI) for interfacing with SYMPHONY and CLP.

What Is Implemented

MibS is an open source solver for bilevel integer programs built on top of the BLIS layer of CHiPPS. Features include

- Branch and Cut framework for general IBLPs
  - *Warm-starting* during feasibility check.
  - *Dual improvement* of variable bounds.
  - *All classes of cuts*.
  - Several *primal heuristics*.
  - Simple *preprocessing*.
- Specialized methods (primarily cuts) for specific problem classes
  - *Pure binary at the upper level*.
  - *Interdiction problems*.
- Standalone heuristics
  - *Greedy* method for interdiction problems.
  - *Weighted sums* method for general problems.
  - *Stationary point* method for general problems.

MibS is available for download at

`http://github.com/tkralphs/MibS`

Outline

Computational Results

- Computations were done on a set of 140 interdiction problems with the following second-stage problem types.
  - Knapsack problems.
  - Random MILP.
- Computations were done on the COR@L compute cluster consisting of 16-core AMD Opteron 2.0 GHz Debian Linux nodes with 32 Gb of memory.

Computational Results: Interdiction instances

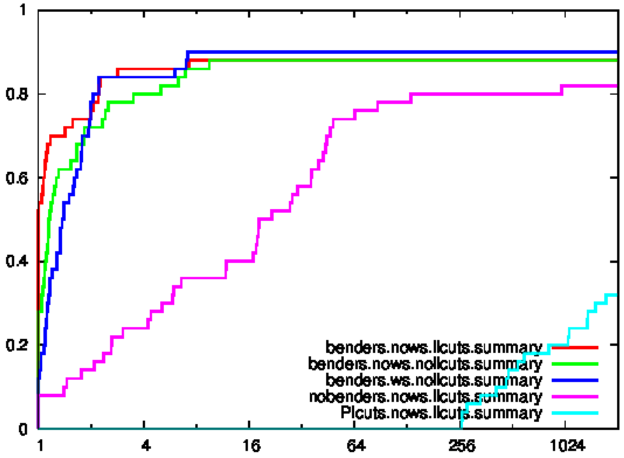


Figure: Performance profile of interdiction instances that can be solved by at least one method with time as a performance measure

Conclusions

- The theory underlying these algorithms is maturing.
- Many of the computational tools necessary for experimentation now also exist.
- Computationally, we have looked primarily at the recourse (stochastic programming) and interdiction (zero sum) cases.
- Substantial progress has been made on these, but they appear much easier than the general case.
- In future work, we plan to leverage what we've learned to develop methodology for the general case.

Shameless Promotion: Free Stuff!

- CHiPPS: Parallel tree search framework
- DIP/DipPy: Decomposition-based modeling language and MILP solver
- DiSCO, OsiConic, CglConic: Mixed integer conic solver
- MibS: Mixed integer bilevel solver
- SYMPHONY: MILP solver framework with bicriteria, warm starting, etc.
- GiMPy, GrUMPy: Visualizations and illustrative implementations for graph and optimization algorithms.
- CuPPy: Cutting planes in Python
- Value Function: Algorithm for constructing value functions
- And more...

`http://github.com/tkralphs`

References I

S. Ahmed, M. Tawarmalani, and N.V. Sahinidis. A finite branch-and-bound algorithm for two-stage stochastic integer programs. *Mathematical Programming*, 100(2): 355–377, 2004.

C.E. Blair. A closed-form representation of mixed-integer program value functions. *Mathematical Programming*, 71:127–136, 1995.

C.E. Blair and R.G. Jeroslow. The value function of a mixed integer program: I. *Discrete Mathematics*, 19:121–138, 1977.

C.E. Blair and R.G. Jeroslow. The value function of an integer program. *Mathematical Programming*, 23:237–273, 1982.

A. Caprara, M. Carvalho, A. Lodi, and G.J. Woeginger. Bilevel knapsack with interdiction constraints. Technical Report OR-14-4, University of Bologna, 2014a.

A. Caprara, M. Carvalho, A. Lodi, and G.J. Woeginger. A study on the computational complexity of the bilevel knapsack problem. *SIAM Journal on Optimization*, 24: 823–838, 2014b.

M. Caramia and R. Mari. Enhanced exact algorithms for discrete bilevel linear problems. *Optimization Letters*, ??:??–??, 2013.

References II

C.C. Carøe and R. Schultz. Dual decomposition in stochastic integer programming. *Operations Research Letters*, 24(1):37–46, 1998.

C.C. Carøe and J. Tind. A cutting-plane approach to mixed 0-1 stochastic integer programs. *European Journal of Operational Research*, 101(2):306–316, 1997.

C.C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming*, 83(1):451–464, 1998.

S DeNegre. *Interdiction and Discrete Bilevel Linear Programming*. Phd, Lehigh University, 2011. URL `http://coral.ie.lehigh.edu/{~}ted/files/papers/ScottDeNegreDissertation11.pdf`.

S DeNegre and T K R. A branch-and-cut algorithm for bilevel integer programming. In *Proceedings of the Eleventh INFORMS Computing Society Meeting*, pages 65–78, 2009. doi: 10.1007/978-0-387-88843-9_4. URL `http://coral.ie.lehigh.edu/~ted/files/papers/BILEVEL08.pdf`.

S.T. DeNegre, T.K. R, and S.A. Tahernejad. Mibs: An open source solver for mixed integer bilevel optimization problems. Technical report, COR@L Laboratory, Lehigh University, 2016.

References III

M. Fischetti, I. Ljubic, M. Monaci, and M. Sinnl. Intersection cuts for bilevel optimization. In *Proceedings of the 18th Conference on Integer Programming and Combinatorial Optimization*, 2016.

Dinakar Gade, Simge Küçükyavuz, and Suvrajeet Sen. Decomposition algorithms with parametric gomory cuts for two-stage stochastic integer programs. *Mathematical Programming*, pages 1–26, 2012.

G. Gamrath, B. Hiller, and J. Witzig. Reoptimization techniques for mip solvers. In *Proceedings of the* 14*th* *International Symposium on Experimental Algorithms*, 2015.

M Güzelsoy. *Dual Methods in Mixed Integer Linear Programming*. Phd, Lehigh University, 2009. URL `http://coral.ie.lehigh.edu/{~}ted/files/papers/MenalGuzelsoyDissertation09.pdf`.

M Güzelsoy and T K R. Duality for mixed-integer linear programs. *International Journal of Operations Research*, 4:118–137, 2007. URL `http://coral.ie.lehigh.edu/~ted/files/papers/MILPD06.pdf`.

References IV

A Hassanzadeh and T K R. A generalized benders' algorithm for two-stage stochastic program with mixed integer recourse. Technical report, COR@L Laboratory Report 14T-005, Lehigh University, 2014a. URL `http://coral.ie.lehigh.edu/~ted/files/papers/SMILPGenBenders14.pdf`.

A Hassanzadeh and T K R. On the value function of a mixed integer linear optimization problem and an algorithm for its construction. Technical report, COR@L Laboratory Report 14T-004, Lehigh University, 2014b. URL `http://coral.ie.lehigh.edu/~ted/files/papers/MILPValueFunction14.pdf`.

Mehdi Hemmati and Cole Smith. A mixed integer bilevel programming approach for a competitive set covering problem. Technical report, Clemson University, 2016.

Robert G Jeroslow. Minimal inequalities. *Mathematical Programming*, 17(1):1–15, 1979.

Ellis L Johnson. Cyclic groups, cutting planes and shortest paths. *Mathematical programming*, pages 185–211, 1973.

References V

Ellis L Johnson. On the group problem for mixed integer programming. In *Approaches to Integer Programming*, pages 137–179. Springer, 1974.

Ellis L Johnson. On the group problem and a subadditive approach to integer programming. *Annals of Discrete Mathematics*, 5:97–112, 1979.

P. Kleniati and C. Adjiman. Branch-and-sandwich: A deterministic global optimization algorithm for optimistic bilevel programming problems. part i: Theoretical development. *Journal of Global Optimization*, 60:425–458, 2014a.

P. Kleniati and C. Adjiman. Branch-and-sandwich: A deterministic global optimization algorithm for optimistic bilevel programming problems. part ii: Convergence analysis and numerical results. *Journal of Global Optimization*, 60: 459–481, 2014b.

N. Kong, A.J. Schaefer, and B. Hunsaker. Two-stage integer programs with stochastic right-hand sides: a superadditive dual approach. *Mathematical Programming*, 108 (2):275–296, 2006.

References VI

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

A Lodi, T K R, and G Woeginger. Bilevel programming and the separation problem. *Mathematical Programming*, 148:437–458, 2014. doi: 10.1007/s10107-013-0700-x. URL `http://coral.ie.lehigh.edu/~ted/files/papers/BilevelSeparation12.pdf`.

L. Lozano and J.C. Smith. A backward sampling framework for interdiction problems with fortification. Technical report, Clemson University, 2016.

A. Mitsos. Global solution of nonlinear mixed integer bilevel programs. *Journal of Global Optimization*, 47:557–582, 2010.

J.T. Moore and J.F. Bard. The mixed integer linear bilevel programming problem. *Operations Research*, 38(5):911–921, 1990.

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

References VII

T K R and M Güzelsoy. The symphony callable library for mixed-integer linear programming. In *Proceedings of the Ninth INFORMS Computing Society Conference*, pages 61–76, 2005. doi: 10.1007/0-387-23529-9_5. URL `http://coral.ie.lehigh.edu/~ted/files/papers/SYMPHONY04.pdf`.

T K R and M Güzelsoy. Duality and warm starting in integer programming. In *The Proceedings of the 2006 NSF Design, Service, and Manufacturing Grantees and Research Conference*, 2006. URL `http://coral.ie.lehigh.edu/~ted/files/papers/DMII06.pdf`.

R. Schultz, L. Stougie, and M.H. Van Der Vlerk. Solving stochastic programs with integer recourse by enumeration: A framework using Gröbner basis. *Mathematical Programming*, 83(1):229–252, 1998.

S. Sen and J.L. Higle. The C3 theorem and a D2 algorithm for large scale stochastic mixed-integer programming: Set convexification. *Mathematical Programming*, 104 (1):1–20, 2005. ISSN 0025-5610.

S. Sen and H.D. Sherali. Decomposition with branch-and-cut approaches for two-stage stochastic mixed-integer programming. *Mathematical Programming*, 106(2):203–223, 2006. ISSN 0025-5610.

References VIII

Hanif D Sherali and J Cole Smith. Two-stage stochastic hierarchical multiple risk problems: models and algorithms. *Mathematical programming*, 120(2):403–427, 2009.

H.D. Sherali and B.M.P. Fraticelli. A modification of Benders' decomposition algorithm for discrete subproblems: An approach for stochastic programs with integer recourse. *Journal of Global Optimization*, 22(1):319–342, 2002.

H.D. Sherali and X. Zhu. On solving discrete two-stage stochastic programs having mixed-integer first-and second-stage variables. *Mathematical Programming*, 108 (2):597–616, 2006.

Andrew C Trapp, Oleg A Prokopyev, and Andrew J Schaefer. On a level-set characterization of the value function of an integer program and its application to stochastic programming. *Operations Research*, 61(2):498–511, 2013.

L.A. Wolsey. Integer programming duality: Price functions and sensitivity analysis. *Mathematical Programming*, 20(1):173–195, 1981. ISSN 0025-5610.

P. Xu. *Three Essays on Bilevel Optimization and Applications*. PhD thesis, Iowa State University, 2012.

References IX

Yang Yuan and Suvrajeet Sen. Enhanced cut generation methods for decomposition-based branch and cut for two-stage stochastic mixed-integer programs. *INFORMS Journal on Computing*, 21(3):480–487, 2009.