Multistage Integer Programming: Algorithms and Complexity

Ted Ralphs
Joint work with Aykut Bulut, Scott DeNegre, Menal Güzelsoy, Anahita Hassanzadeh

1 COR@L Lab, Department of Industrial and Systems Engineering, Lehigh University
2 SAS Institute, Advanced Analytics, Operations Research R & D
3 The Chartis Group

University of Houston, March 4, 2014
1. Introduction

2. Value Function

3. Algorithm

4. Conclusions
A Bit of Game Theory

The optimization problems we address can be conceptualized as *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 *multistage* optimization problem.
- The move must be determined by taking into account the *uncertainty about future stages*.
Multilevel and Multistage Games

- In the literature, the term *multilevel* is used for competitive games in which there is no chance player.
- *Multistage* is used 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.
Quick Examples

Cost Components

Scheduled

Actual

Idle Time  Wait Time  Overtime

OR opening time (7:00)  OR closing time (4:00)

Cost = C_W \text{(waittime)} + C_I \text{(idletime)} + C_O \text{(overtime)}
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 $\iff$ multiple objectives, single DM
- Mathematical Programming with Recourse $\iff$ multiple stages, single DM
- Multilevel Programming $\iff$ 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 arise in a remarkably wide array of 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
With two stages, we have the following general formulation:

\[
z_{2SMILP} = \min_{x \in P_1} \Psi(x) = \min_{x \in P_1} \left\{ c^\top x + \Xi(x) \right\},
\]

where

\[
P_1 = \{ x \in X \mid Ax = b, x \geq 0 \}
\]

is the first-stage feasible region with \( X = \mathbb{Z}_{+}^{r_1} \times \mathbb{R}_{+}^{n_1-r_1} \).

\( \Xi \) represents the leader’s expectation of the impact of future uncertainty.

The canonical form employed in stochastic programming with recourse is

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

\( \phi \) is the second-stage value function to be defined shortly.

\( T_\omega \in \mathbb{Q}^{m_2 \times n_1} \) and \( h_\omega \in \mathbb{Q}^{m_2} \) represent the input to the second-stage problem for scenario \( \omega \in \Omega \).
The Second-Stage Value Function

- The structure of the objective function $\Psi$ depends primarily on the structure of the value function

$$
\phi(\beta) = \min \left\{ d^\top y \mid y \in \arg\min_{y \in P_L(\beta)} q^\top y \right\}. 
$$

where

$$
P_2(\beta) = \{ y \in Y \mid Wy = \beta \}
$$

is the second-stage feasible region with respect to a given right-hand side $\beta$ and $Y = \mathbb{Z}_+^{r_2} \times \mathbb{R}_+^{n_2-r_2}$.

- The second-stage problem is parameterized on the unknown value $\beta$ of the right-hand side.
- This value is determined jointly by the realized value of $\omega$ and the values of the first-stage decision variables.
- The second-stage solution is evaluated with respect to two objective vectors, $q$ and $d$, that represent the (possibly) differing valuations of the two players.
Two-Stage Stochastic Program with Recourse

For the remainder of the talk, we consider the simpler case of two-stage stochastic programming:

\[
\begin{align*}
\min \Psi(x) &= \min_{x \in P_1} c^T x + \sum_{\omega \in \Omega} p_\omega \phi(h_\omega - T_\omega x) \quad \text{(SP)} \\
\phi(\beta) &= \min_{y \in P_2(\beta)} q^T y \quad \text{(RP)}
\end{align*}
\]

where

\[
\phi(\beta) = \min_{y \in P_2(\beta)} q^T y
\]

In this talk, we assume

- \(\omega\) follows a discrete distribution with a finite support,
- \(W\) and \(q\) are fixed,
- \(P_1\) is compact, and
- \(E_{\omega \in \Omega}[\phi(h_\omega - T_\omega x)]\) is finite for all \(x \in X\).

Unless otherwise indicated, all probability distributions will be uniform.
Illustrating the Value Function

**Example 1**

\[ \phi(\beta) = \min 6y_1 + 4y_2 + 3y_3 + 4y_4 + 5y_5 + 7y_6 \]

\[ s.t. 2y_1 + 5y_2 - 2y_3 - 2y_4 + 5y_5 + 5y_6 = \beta \]

\[ y_1, y_2, y_3 \in \mathbb{Z}_+, y_4, y_5, y_6 \in \mathbb{R}_+. \]
Illustrating the Objective Function

Example 2

\[ \Psi(x) = -3x_1 - 4x_2 + \sum_{\omega \in \Omega} \phi(h_\omega - 2x_1 - 0.5x_2) \]  

(Ex.SMP)

and \( \Omega = \{1, 2\}, h_1 = 6, h_2 = 12. \)

Note the similarity in structure of the objective function to the value function.
MILP Value Function (Pure Integer)

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

Example 3

\[ \phi(b) = \min 3x_1 + \frac{7}{2}x_2 + 3x_3 + 6x_4 + 7x_5 + 5x_6 \]
\[ \text{s.t. } 6x_1 + 5x_2 - 4x_3 + 2x_4 - 7x_5 + x_6 = b \]
\[ x_1, x_2, x_3, x_4, x_5, x_6 \in \mathbb{Z}^+ \]
Example 4

\[ \phi(b) = \min 3x_1 + \frac{7}{2}x_2 + 3x_3 + 6x_4 + 7x_5 + 5x_6 \]

\[ \text{s.t. } 6x_1 + 5x_2 - 4x_3 + 2x_4 - 7x_5 + x_6 = b \]

\[ x_1, x_2, x_3 \in \mathbb{Z}_+, x_4, x_5, x_6 \in \mathbb{R}_+ \]
Consider the general form of the second-stage value function

$$\phi(\beta) = \min q_I^\top y_I + q_C^\top y_C$$

s.t. $W_I y_I + W_C y_C = b,$

$y \in \mathbb{Z}^{r_2}_+ \times \mathbb{R}^{n_2-r_2}_+$

(MILP)

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

$$\phi_C(\beta) = \min q_C^\top y_C$$

s.t. $W_C y_C = \beta,$

$y_c \in \mathbb{R}^{n_2-r_2}_+$

(CR)

and the similarly defined *integer restriction*:

$$\phi_I(\beta) = \min q_I^\top y_I$$

s.t. $W_I y_I = \beta$

$y_I \in \mathbb{Z}^{r_2}_+$

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

$$\phi(\beta) = \min q_I^\top y_I + \phi_C(\beta - W_I y_I)$$

s.t. $y_I \in \mathbb{Z}_{r_2}^+$ \hspace{1cm} (6)

- From this we see that the value function is comprised of the minimum of a set of shifted copies of $\phi_C$.

- The set of shifts, along with $\phi_C$ describe the value function exactly.

- For $\hat{y}_I \in \mathbb{Z}_{r_2}^+$, let

$$\phi_C(\beta, \hat{y}_I) = q_I^\top \hat{y}_I + \phi_C(\beta - W_I \hat{y}_I) \ \forall \beta \in \mathbb{R}^{m_2}. \hspace{1cm} (7)$$

- Then we have that $\phi(\beta) = \min_{y_I \in \mathbb{Z}_{r_2}^+} \phi_C(\beta, \hat{y}_I)$. 
Illustrating the Continuous Restriction

Example 5

\[ \phi_C(\beta) = \min 6y_1 + 7y_2 + 5y_3 \]

s.t. \[ 2y_1 - 7y_2 + y_3 = \beta \]

\[ y_1, y_2, y_3 \in \mathbb{R}_+ \]
Value Function of the Continuous Restriction

Recall the previously defined continuous restriction.

\[ \phi_C(\beta) = \min q_C^\top y_C \]
\[ \text{s.t. } W_C y_C = \beta \]
\[ y_C \in \mathbb{R}_+^n \]  \hspace{2cm} (CR)

When the dual of (CR) is feasible, the epigraph of \( \phi_C \) is the convex cone

\[ \mathcal{L} := \text{cone}\{(W_1, q_1), (W_2, q_2), \ldots, (W_n, q_n), (0, 1)\} \]  \hspace{2cm} (8)

Let \( u_1, \ldots, u_k \) be extreme points of the feasible region of the dual of (CR) and \( d_1, \ldots, d_p \) be its extreme directions. Then

\[ \mathcal{L} := \{(\beta, z) : z \geq u_i^\top \beta, i = 1, \ldots, k, d_j^\top \beta \leq 0, j = 1, \ldots, p\}. \]  \hspace{2cm} (9)
Properties of MILP Value Function

- We can improve on the previous representation by deriving a *minimal* discrete set that suffices to describe $\phi$.

**Theorem 1** [Hassanzadeh et al., 2014]

*Under the assumption that $\{\beta \in \mathbb{R}^{m_2} | \phi_I(\beta) < \infty\}$ is finite, there exists a finite set $S \subseteq Y$ such that*

$$
\phi(\beta) = \min_{y_I \in S} \{q_I^T y_I + \phi_C(\beta - W_I y_I)\}.
$$

- The points in $S$ are the points of *strict local convexity* of the value function.
- Associated with each of these points is a region (the *local stability set*) over which the integer part of the optimal solution remains constant.
- The value function of the MILP, when restricted to that region, is a translation of the value function of the continuous restriction (and thus convex).
- In [Hassanzadeh et al., 2014], we describe an algorithm for constructing a superset of $S$ that is easy to implement.
Points of Strict Local Conexity

Example 6

The figure above shows the points of strict local convexity and the associated local stability sets for the previous example.
Outline

1. Introduction
2. Value Function
3. Algorithm
4. Conclusions
Benders’ Principle (Linear Programming)

\[ z_{LP} = \min_{(x,y) \in \mathbb{R}^n} \{ c'x + c''y \mid A'x + A''y \geq b \} \]

\[ = \min_{x \in \mathbb{R}^n} \{ c'x + \phi(b - A'x) \}, \]

where

\[ \phi(d) = \min c''y \]

\[ \text{s.t. } A''y \geq d \]

\[ y \in \mathbb{R}^{n''} \]

**Basic Strategy:**

- The function \( \phi \) is the *value function* of a linear program.
- The value function is piecewise linear and convex.
- We iteratively generate a lower approximation by sampling the domain.
Example

\[
z_{LP} = \min \quad x + y
\]

s.t. \quad 25x - 20y \geq -30
\quad -x - 2y \geq -10
\quad -2x + y \geq -15
\quad 2x + 10y \geq 15
\quad x, y \in \mathbb{R}

Ralphs, et al. (COR@L Lab)
Value Function Reformulation

\[ z_{LP} = \min_{x \in \mathbb{R}} x + \phi(x), \]

where

\[ \phi(x) = \min \ y \]

\[ \text{s.t.} \quad -20y \geq -30 - 25x \]
\[ -2y \geq -10 + x \]
\[ y \geq -15 + 2x \]
\[ 10y \geq 15 - 2x \]
\[ y \in \mathbb{R} \]
Benders’ Principle (Integer Programming)

\[
\begin{align*}
\z_{\text{IP}} &= \min_{(x,y) \in \mathbb{Z}^n} \left\{ c'x + c''y \mid A'x + A''y \geq b \right\} \\
&= \min_{x \in \mathbb{R}^n} \left\{ c'x + \phi(b - A'x) \right\},
\end{align*}
\]

where

\[
\phi(d) = \min c''y \\
\text{s.t. } A''y \geq d \\
y \in \mathbb{Z}^{n''}
\]

Basic Strategy:

- Here, \( \phi \) is the value function of an integer program.
- In the general case, the function \( \phi \) is piecewise linear but not convex.
- Here, we also iteratively generate a lower approximation by evaluating \( \phi \).
Example

\[ z_{IP} = \min \quad x + y \]
\[
\text{s.t.} \quad 25x - 20y \geq -30 \\
-2x + y \geq -15 \\
2x + 10y \geq 15
\]
\[ x, y \in \mathbb{Z} \]
Value Function Reformulation

\[ z_{IP} = \min_{x \in \mathbb{Z}} x + \phi(x), \]

where

\[ \phi(x) = \min \quad y \]
\[ \text{s.t.} \quad -20y \geq -30 - 25x \]
\[ -2y \geq -10 + x \]
\[ y \geq -15 + 2x \]
\[ 10y \geq 15 - 2x \]
\[ y \in \mathbb{Z} \]
Related Algorithms

The algorithmic framework we utilize builds on a number of previous works.

  - Linear cuts in first stage for binary first stage
  - Optimality cuts from B&B and cutting plane, applied to pure integer second stage
  - Disjunctive programming approaches and cuts in the second stage
- Value function approaches: Pure integer case [Ahmed et al., 2004, Kong et al., 2006]
- Scenario decomposition [Carøe and Schultz, 1998]
- Enumeration/Gröbner basis reduction [Schultz et al., 1998]
### Summary of Related Work

<table>
<thead>
<tr>
<th></th>
<th>First Stage</th>
<th>Second Stage</th>
<th>Stochasticity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R</td>
<td>Z</td>
<td>B</td>
</tr>
<tr>
<td>[Laporte and Louveaux, 1993]</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Carøe and Tind, 1997]</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>[Carøe and Tind, 1998]</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>[Carøe and Schultz, 1998]</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>[Schultz et al., 1998]</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Sherali and Fraticelli, 2002]</td>
<td>*</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>[Ahmed et al., 2004]</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>[Sen and Higle, 2005]</td>
<td>*</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>[Sen and Sherali, 2006]</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>[Sherali and Zhu, 2006]</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>[Kong et al., 2006]</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>[Sherali and Smith, 2009]</td>
<td>*</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>[Yuan and Sen, 2009]</td>
<td>*</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>[Ntaiimo, 2010]</td>
<td>*</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>[Gade et al., 2012]</td>
<td>*</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>[Trapp et al., 2013]</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>Current work</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

Ralphs, et al. (COR@L Lab)  
Multilevel Integer Programming  
31 / 55
We already observed that for an effective integer Benders’ method, we need effective lower bounding functions to approximate the MILP value function.
A dual function \( \varphi : \mathbb{R}^{m_2} \rightarrow \mathbb{R} \cup \{\pm \infty\} \) is

\[
\varphi(\beta) \leq \phi(\beta) \quad \forall \beta \in \mathbb{R}^{m_2}
\]  

(11)

For a particular instance \( \hat{\beta} \), the dual problem is

\[
\phi_D = \max\{\varphi(\hat{\beta}) : \varphi(\beta) \leq \phi(\beta) \quad \forall \beta \in \mathbb{R}^{m_2}, \varphi : \mathbb{R}^{m_2} \rightarrow \mathbb{R} \cup \{\pm \infty\}\}
\]

(12)

Let \( \mathcal{F} \) be a set of dual functions generated so far. Then Benders’ master problem is

\[
\begin{align*}
\min & \quad c^T x + \theta \\
\text{subject to} & \quad \theta \geq \sum_{\omega \in \Omega} \max_{f \in \mathcal{F}} f(h_\omega - T_\omega x) \\
x & \in \mathcal{P}_1
\end{align*}
\]

(MP)
Let $T$ be set of the terminating nodes of the tree. Then in a terminating node $t \in T$ we solve:

$$\min c^\top x$$

s.t. $Ax = b,$

$$l^t \leq x \leq u^t, x \geq 0$$

The dual at node $t$:

$$\max \{\pi^t b + \underline{\pi}^t l^t + \bar{\pi}^t u^t\}$$

s.t. $\pi^t A + \underline{\pi}^t + \bar{\pi}^t \leq c^\top$

$$\pi \geq 0, \underline{\pi} \leq 0$$

We obtain the following strong dual function:

$$\min_{t \in T}\{\pi^t b + \underline{\pi}^t l^t + \bar{\pi}^t u^t\}$$
Warm Starting the Solution Process

- Here, we illustrate the procedure.
- We can improve on the basic scheme by warm starting the solution of each subproblem from the tree generated during solution of the previous subproblem.
Warm Starting the Solution Process

- Here, we illustrate the procedure.
- We can improve on the basic scheme by warm starting the solution of each subproblem from the tree generated during solution of the previous subproblem.
Warm Starting the Solution Process

- Here, we illustrate the procedure.
- We can improve on the basic scheme by warm starting the solution of each subproblem from the tree generated during solution of the previous subproblem.
Warm Starting the Solution Process

- Here, we illustrate the procedure.
- We can improve on the basic scheme by warm starting the solution of each subproblem from the tree generated during solution of the previous subproblem.
Generating the Value Function in a Single Tree

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

\[
\phi^*(\beta) = \min_{t \in T} q_{I_t}^T y_{I_t}^t + \phi_{N\setminus I_t}(\beta - W_{I_t} y_{I_t}^t),
\]

(16)

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

**Theorem 2** 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 \(\phi^* = \phi\).
Example of Value Function Tree

Node 0

\[ y_3 = 0 \]

\[ y_3 \geq 1 \]

Node 1

\[ y_2 = 0 \]

\[ y_2 \geq 1 \]

Node 2

\[ \text{max}\{\beta - 1, \beta\} \]

\[ y_2 = 1 \]

\[ y_2 \geq 2 \]

Node 3

\[ \text{max}\{\beta + 28, \beta - 2\} \]

\[ -2\beta + 42 \]

Node 4

\[ \text{max}\{\beta + 6, g_7 = -2\beta - 1\} \]

\[ y_3 = 1 \]

\[ y_3 \geq 2 \]

Node 5

\[ y_3 \geq 3 \]

Node 6

\[ \text{max}\{2\beta + 28, \beta - 2\} \]

\[ -2\beta + 42 \]

Node 7

\[ \text{max}\{\beta + 10, g_9 = -2\beta - 2\} \]

\[ y_3 = 2 \]

\[ y_3 \geq 3 \]

Node 8

\[ y_3 \geq 4 \]

Node 9

\[ \text{max}\{\beta + 15, -2\beta - 3\} \]

\[ y_3 = 3 \]

\[ y_3 \geq 5 \]

Node 10

\[ y_3 \geq 6 \]

Node 11

\[ \text{max}\{\beta + 20, -2\beta - 5\} \]

\[ \beta + 30 \]

Node 12

\[ y_3 = 4 \]

\[ y_3 \geq 5 \]

Node 13

\[ y_3 = 5 \]

\[ y_3 \geq 6 \]

Node 14

\[ y_3 \geq 7 \]

Node 15

\[ \beta + 14, \beta \]

\[ y_3 \geq 8 \]

Node 16

\[ y_3 \geq 9 \]

Node 17

\[ \beta + 10, g_9 = -2\beta - 2\]
Example of Value Function Tree

\[ \Phi_{\text{MILP}}(\beta) \]

Leveling:
- Node 17
- Node 15
- Node 13
- Node 11
- Node 9
- Node 2
- Node 4
- Node 6

\( \beta \) Scale:
- -10
- -8
- -6
- -4
- -2
- 2
- 4
- 6
- 8
- 10
Master Problem Formulation

Notation:

- $s, r \in \{1, \ldots, S\}$ where $S$ is the number of scenarios
- $p \in \{1, \ldots, k\}$ where $k$ is the iteration number
- $n \in \{1, \ldots, N(p, r)\}$ where $N(p, r)$ is the number of terminating nodes in the B&B tree solved for scenario $r$ at iteration $p$.
- $\theta_s = \mathcal{F}(h(s) - \beta)$
- $t_{spr} = F^p_r(h(s) - \beta)$ the approximation of scenario $s$’s recourse obtained from the optimal dual function of iteration $p$ and scenario $r$.
- $\nu_{prn}, a_{prn}$ respectively, the dual vector and intercept obtained from node $n$ of the B&B tree solved for scenario $r$ in iteration $p$.
- $p_s$ probability of scenario $s$
- $M > 0$ an appropriate large number
\[
f^k = \min c^\top x + \sum_{s=1}^{S} p_s \theta_s
\]

s.t. \( \theta_s \geq t_{spr} \)

\[
t_{spr} \leq a_{prn} + \nu_{prn}^\top (h(s) - T(s)x) \quad \forall s, p, r
\]

\[
t_{spr} \geq a_{prn} + \nu_{prn}^\top (h(s) - T(s)x) - Mu_{sprn} \quad \forall s, p, r, n
\]

\[
\sum_{n=1}^{N} u_{sprn} = N(p, r) - 1 \quad \forall s, p, r
\]

\[
x \in X, u_{sprn} \in \mathbb{B} \quad \forall s, p, r, n
\]
Example

Consider

\[
\min f(x) = \min -3x_1 - 4x_2 + \sum_{s=1}^{2} 0.5Q(x, s)
\]

\[
\text{s.t. } x_1 + x_2 \leq 5
\]

\[
x \in \mathbb{Z}_+
\]

where

\[
Q(x, s) = \min 3y_1 + \frac{7}{2} y_2 + 3y_3 + 6y_4 + 7y_5
\]

\[
\text{s.t. } 6y_1 + 5y_2 - 4y_3 + 2y_4 - 7y_5 = h(s) - 2x_1 - \frac{1}{2}x_2
\]

\[
y_1, y_2, y_3 \in \mathbb{Z}_+, y_4, y_5 \in \mathbb{R}_+
\]

with \(h(s) \in \{-4, 10\}\).
Example
Conclusions

Non-convex optimality cuts are ugly. But they may be worthwhile!

- We have developed an algorithm for the two-stage problem with general mixed integer in both stages.
- The algorithm uses the Benders’ framework with B&B dual functions as the optimality cuts.
- Such cuts have computationally desirable properties such as warm-starting.
- We need to keep the size of approximations small. This can be done through warm-starting trees and scenario bunching.
Future Work

- We have implemented the algorithm using SYMPHONY as our mixed-integer linear optimization solver.
- Warm-starting a B&B tree is possible in the solver.
- We so far have a fairly “naive” implementation and anticipate much improvement is possible.
- In particular, we should be able to exploit parallelism much more easily here than in the traditional MILP case.
- We also need to develop a scenario bunching scheme. Doing this, we decide on the local area of the tree to examine.
- Finally, we hope to move on soon to the more general case of multilevel programming.


