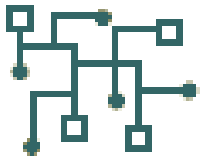
Generation and Representation of Piecewise Polyhedral Value Functions

Ted Ralphs[1]
Joint work with Menal Güzelsoy[2] and Anahita Hassanzadeh[1]

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

MINLP Workshop, Carnegie Mellon University, June 2, 2014



ISE Industrial and Systems Engineering

COR@L COMPUTATIONAL OPTIMIZATION RESEARCH AT LEHIGH

Outline

Outline

Motivation: Two-Stage Mixed Integer Optimization

- We have the following general formulation:

$$z_{\text{2SMILP}} = \min_{x \in \mathcal{P}_1} \Psi(x) = \min_{x \in \mathcal{P}_1} \left\{ c^\top x + \Xi(x) \right\}, \tag{1}$$

  where

$$\mathcal{P}_1 = \{ x \in X \mid Ax = b, x \geq 0 \} \tag{2}$$

  is the *first-stage feasible region* with $X = \mathbb{Z}_+^{r_1} \times \mathbb{R}_+^{n_1 - r_1}$.
- Ξ represents the impact of future uncertainty.
- The canonical form employed in stochastic programming with recourse is

$$\Xi(x) = \mathbb{E}_{\omega \in \Omega} \left[\phi(h_\omega - T_\omega x) \right], \tag{3}$$

- ϕ 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 $\omega \in \Omega$.

This is a MINLP. How do we solve it?

The Second-Stage Value Function

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

$$\phi(\beta) = \min_{y \in \mathcal{P}_2(\beta)} q^\top y \tag{RP}$$

  where

$$\mathcal{P}_2(\beta) = \{y \in Y \mid Wy = \beta\} \tag{4}$$

  is the *second-stage feasible region* with respect to a given right-hand side β and $Y = \mathbb{Z}_+^{r_2} \times \mathbb{R}_+^{n_2 - r_2}$.
- 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.
- We assume
  - ω follows a uniform distribution with a finite support,
  - $\mathcal{P}_1$ is compact, and
  - $\mathbb{E}_{w \in \Omega}[\phi(h_\omega - T_\omega x)]$ is finite for all $x \in X$.

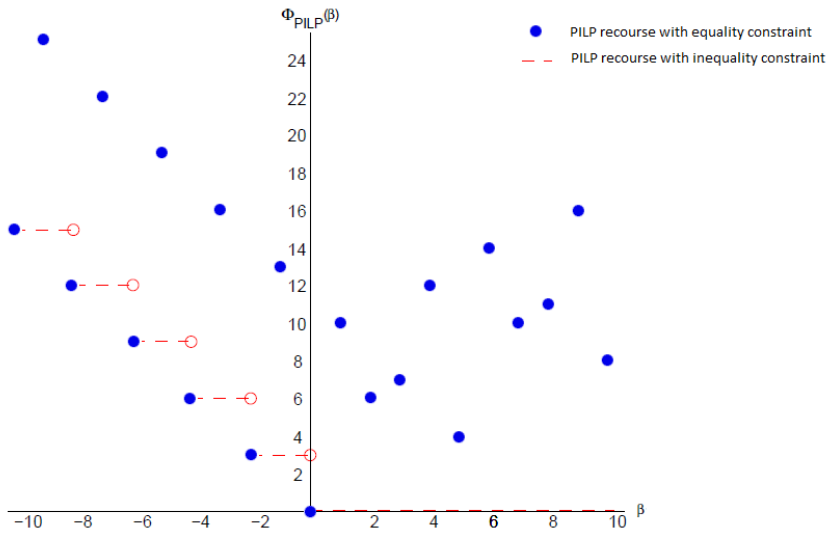
Outline

MILP Value Function (Pure)

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

Example 1

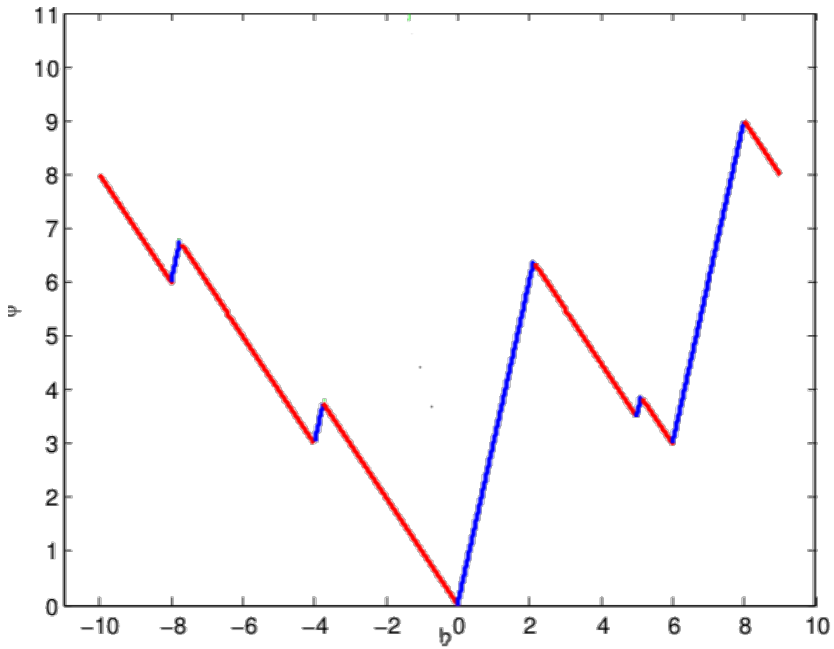
$$\phi(b) = \min\ 3x_1 + \frac{7}{2}x_2 + 3x_3 + 6x_4 + 7x_5 + 5x_6$$
$$s.t.\ 6x_1 + 5x_2 - 4x_3 + 2x_4 - 7x_5 + x_6 = b$$
$$x_1, x_2, x_3, x_4, x_5, x_6 \in \mathbb{Z}_+$$



MILP Value Function (Mixed)

Example 2

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



Continuous and Integer Restriction of an MILP

Consider the general form of the second-stage value function

$$\begin{aligned} \phi(\beta) = \min\ & q_I^\top y_I + q_C^\top y_C \\ \text{s.t. } & W_I y_I + W_C y_C = b, \\ & y \in \mathbb{Z}_+^{r_2} \times \mathbb{R}_+^{n_2 - r_2} \end{aligned} \tag{MILP}$$

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

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

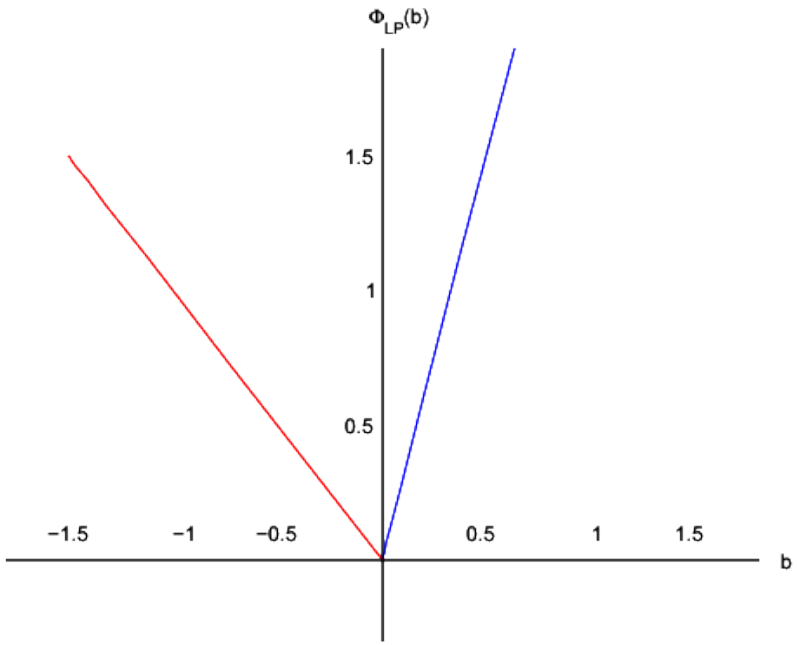
and the similarly defined *integer restriction*:

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

Value Function of the Continuous Restriction

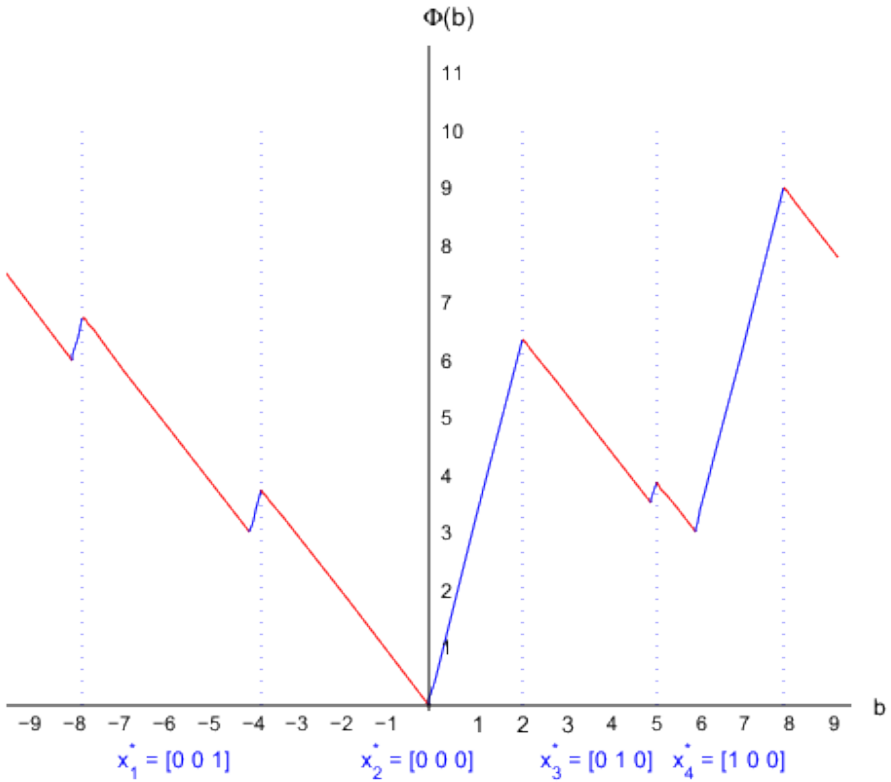
Example 3

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



Points of Strict Local Conexity

Example 4



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

$$\phi(\beta) = \min_{y_I \in \mathcal{S}} \{q_I^\top y_I + \phi_C(\beta - W_I y_I)\}. \tag{5}$$

Outline

Representing the Value Function

In practice, the value function can be represented in primarily two different ways.

1. On the one hand, the value function is uniquely determined by its points of *strict local convexity*.
   - Embedding the value function using this representation involves explicitly listing these points and choosing one (binary variables).
   - The corresponding continuous part of the solution can be generated dynamically or can also be represented explicitly by dual extreme points.
2. The value function can also be represented explicitly in terms of its *polyhedral pieces*.
   - In this case, the points of strict local convexity are implicit and the selection is of the relevant piece or pieces.
   - This yields a much larger representation.

Embedding the Value Function

Algorithmically, the value function can be embedded in a number of ways.

1. If the objective functions of the inner and outer optimizations "agree," we can embed the inner constraints and evaluate the value function implicitly.
   - This amounts to solving the extensive form in the case of stochastic integer programming.
   - For stochastic programs with many scenarios, this form can be too large to solve explicitly.
2. We can generate the value function a priori and then embed the full representation.
   - The full representation might be very large
   - It is likely that most of the representation is irrelevant to the computation.
3. We can dynamically generate "relevant" parts of the representation ala cut generation.

It is also possible to take a hybrid approach in which we, e.g., generate the full representation a priori, but add parts dynamically.

Conceptual Algorithm for Generating the Value Function

Algorithm

Initialize: Let $\bar{z}(b) = \infty$ 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 $\bar{z}(b) = \min\{\bar{z}, \bar{z}(b; x_I^k)\}$ for all $b \in B$.
- $k \leftarrow k + 1$.
- Solve

$$\begin{aligned} \Gamma^k = \max\ & \bar{z}(b) - c_I^\top x_I \\ \text{s.t. } & A_I x_I = b \\ & x_I \in \mathbb{Z}_+^r. \end{aligned} \tag{SP}$$

  to obtain x_I^k.
- Set $S^k \leftarrow S^{k-1} \cup \{x^k\}$

end while
return $z(b) = \bar{z}(b)$ for all $b \in B$.

Conceptual Algorithm for Generating the Value Function

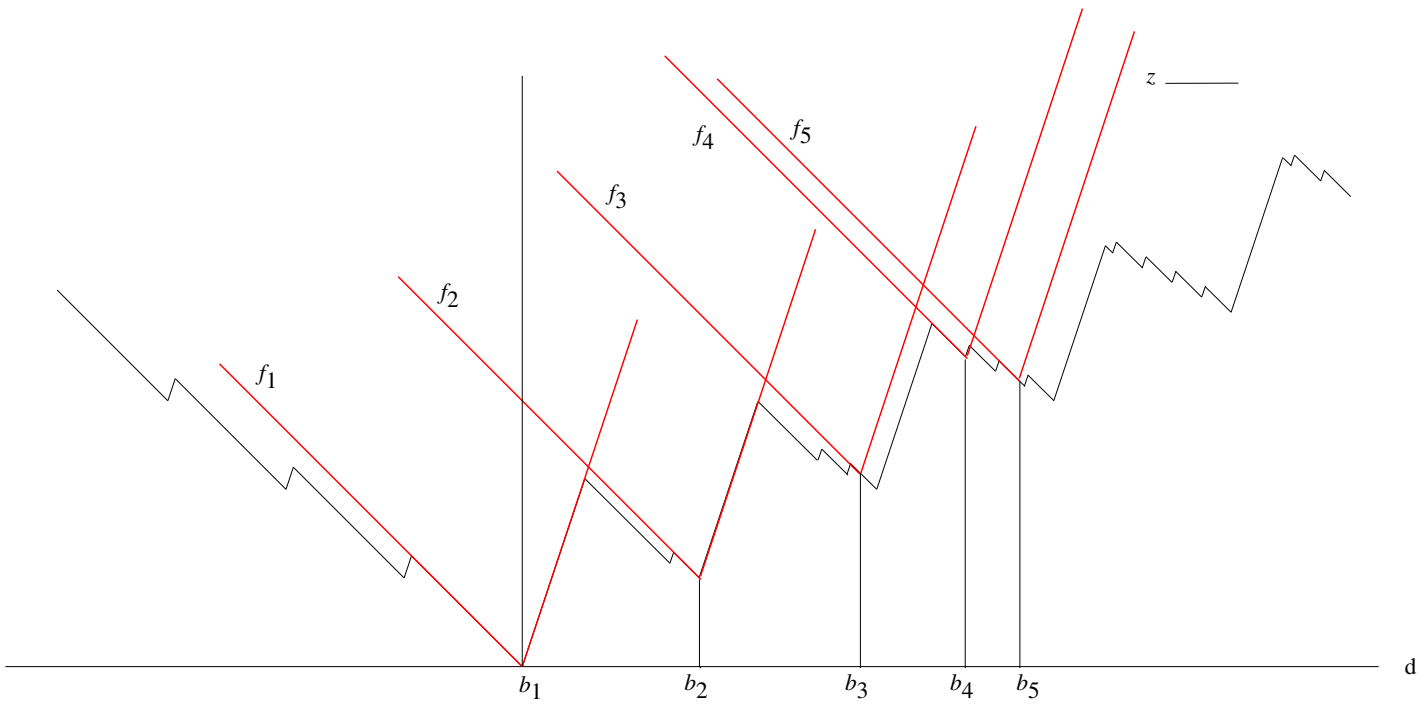


Figure : Upper bounding functions obtained at right-hand sides $b_i, i = 1, \ldots, 5$.

Formulating (SP)

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

$$
\begin{aligned}
\Gamma^k = \max\ & \theta \\
\text{s.t. } & \theta + c_I^\top x_I \le c_I^\top x_I^i + (A_I x_I - A_I x_I^i)^\top \nu^i \quad i = 1, \ldots, k-1 \\
& A_C^\top \nu^i \le c_C \quad i = 1, \ldots, k-1 \\
& \nu^i \in \mathbb{R}^m \quad i = 1, \ldots, k-1 \\
& x_I \in \mathbb{Z}_+^r .
\end{aligned}
\tag{6}
$$

Computational Results

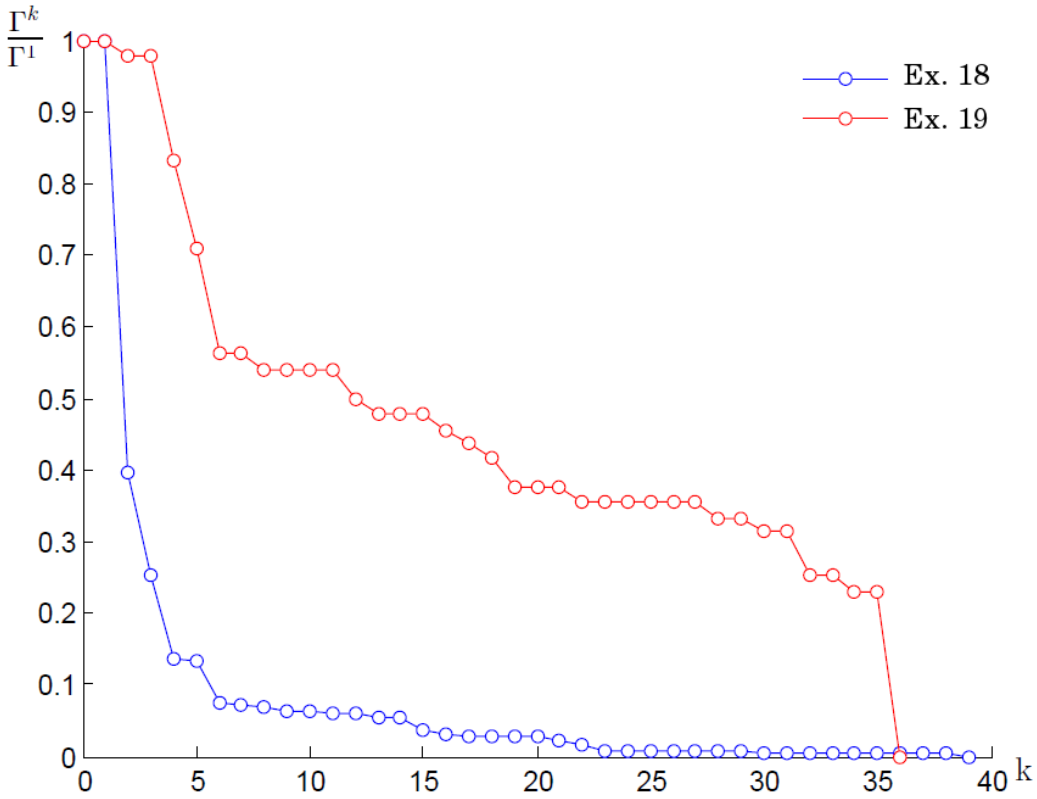
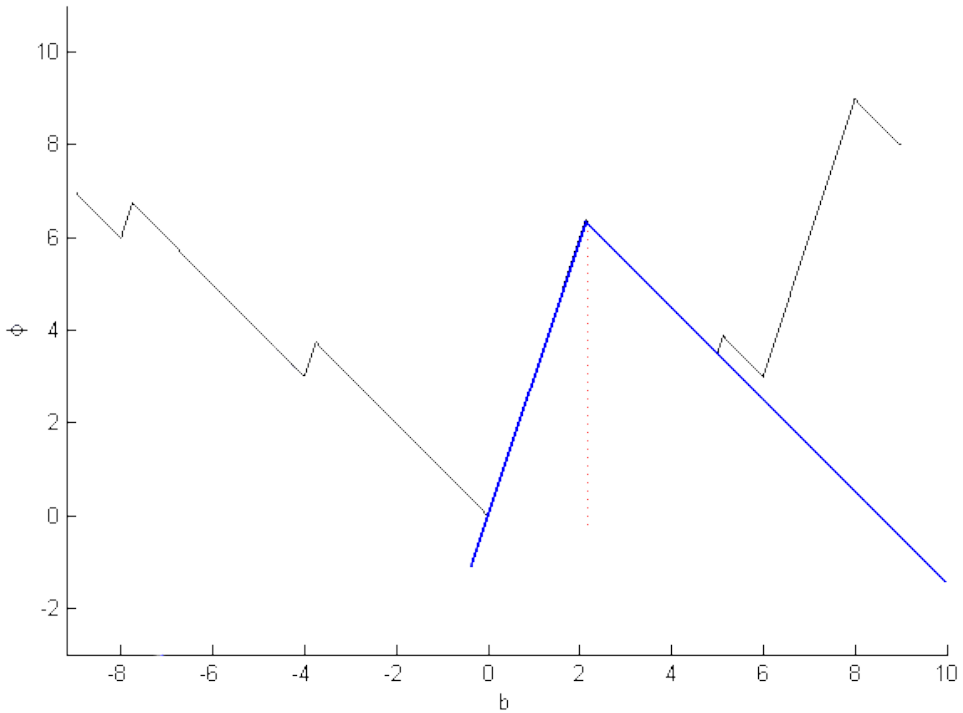


Figure : Normalized approximation gap vs. iteration number.

Generating the Value Function by Branch and Bound

- Our first algorithm was based on refining a single upper approximation (essentially a restriction of the full value function).
- It is also possible to construct the value function by lower approximation using branch and bound.
- Any branch-and-bound tree yields a lower approximation of the value function.



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

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

$$\begin{aligned} &\min c^\top x \\ &\text{s.t. } Ax = b, \\ &\quad l^t \leq x \leq u^t, x \geq 0 \end{aligned} \tag{7}$$

The dual at node t:

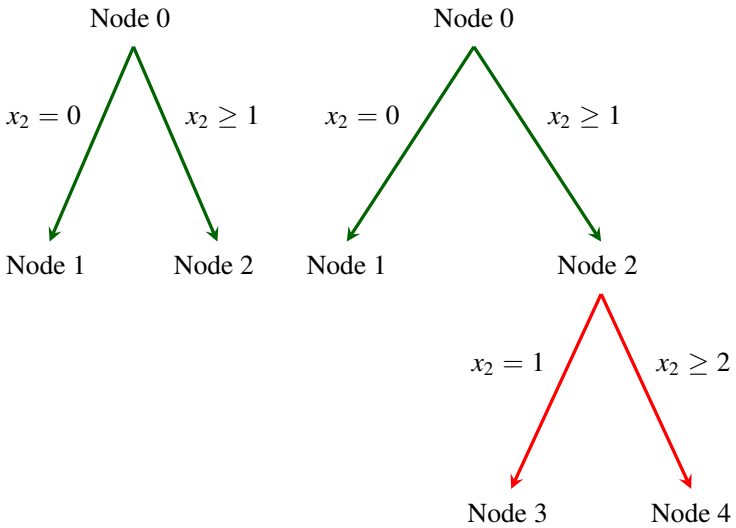
$$\begin{aligned} &\max \{\pi^t b + \underline{\pi}^t l^t + \bar{\pi}^t u^t\} \\ &\text{s.t. } \pi^t A + \underline{\pi}^t + \bar{\pi}^t \leq c^\top \\ &\quad \underline{\pi} \geq 0, \bar{\pi} \leq 0 \end{aligned} \tag{8}$$

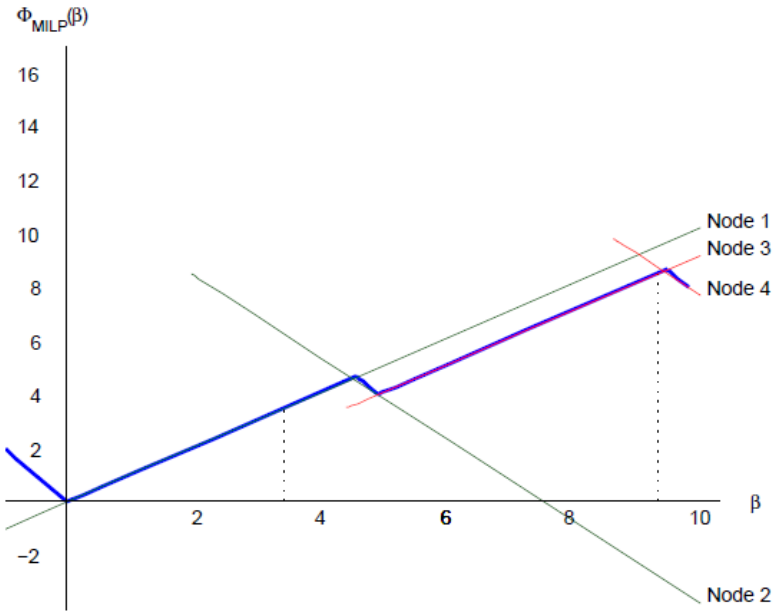
We obtain the following strong dual function:

$$\min_{t \in T} \{\pi^t b + \underline{\pi}^t l^t + \bar{\pi}^t u^t\} \tag{9}$$

Iterative Refinement

- The tree obtained from evaluating $\phi(b)$ yields a dual function strong at b.
- 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.





Tree Representation of the Value Function

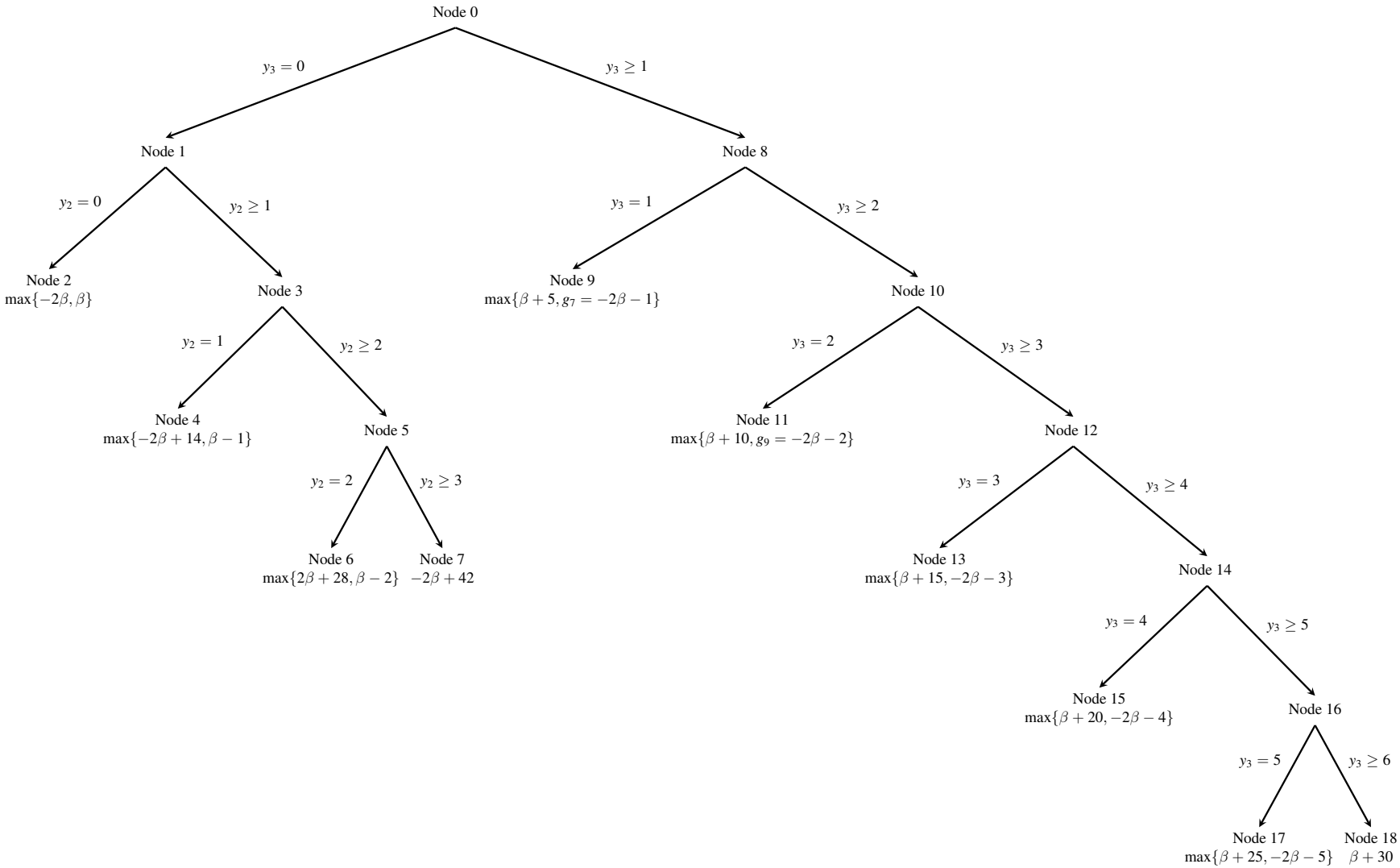
- Continuing the process, we eventually generate the entire value function.
- Consider the strengthened dual

$$\underline{\phi}^*(\beta) = \min_{t \in T} q_{I_t}^\top y_{I_t}^t + \phi_{N \setminus I_t}(\beta - W_{I_t} y_{I_t}^t), \tag{10}$$

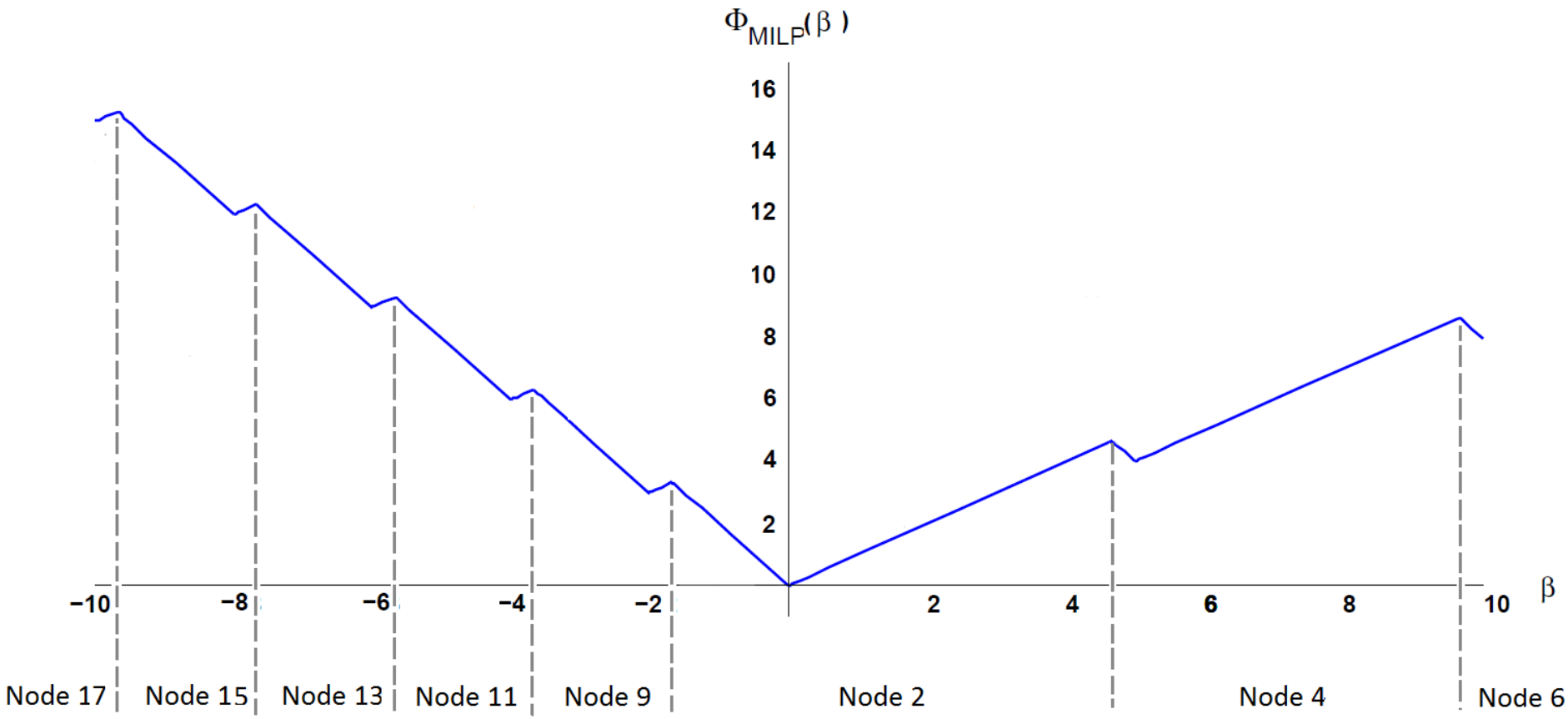
- I_t is the set of indices of fixed variables, $y_{I_t}^t$ are the values of the corresponding variables in node t.
- $\phi_{N \setminus I_t}$ is the value function of the linear program including only the unfixed variables.

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* $\underline{\phi}^* = \phi$.

Example of Value Function Tree



Correspondence of Nodes and Local Stability Regions



Back to Stochastic Programming: Lit Review

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

Benders Master Variables

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.
- $\theta_s = \mathcal{F}(h(s) - \beta)$
- $t_{spr} = F_r^p(h(s) - \beta)$ the approximation of scenario s's recourse obtained from the optimal dual function of iteration p and scenario r.
- ν_{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

Benders Master Formulation

$$
\begin{aligned}
f^k = \min\ & c^\top x + \sum_{s=1}^{S} p_s \theta_s \\
\text{s.t. } & \theta_s \geq t_{spr} && \forall s, p, r \\
& t_{spr} \leq a_{prn} + \nu_{prn}^\top (h(s) - T(s)x) && \forall s, r, p, n \\
& t_{spr} \geq a_{prn} + \nu_{prn}^\top (h(s) - T(s)x) - M u_{sprn} && \forall s, p, r, n \\
& \sum_{n=1}^{N} u_{sprn} = N(p, r) - 1 && \forall s, p, r \\
& x \in X, u_{sprn} \in \mathbb{B} && \forall s, p, r, n
\end{aligned}
\tag{master}
$$

Example

Consider

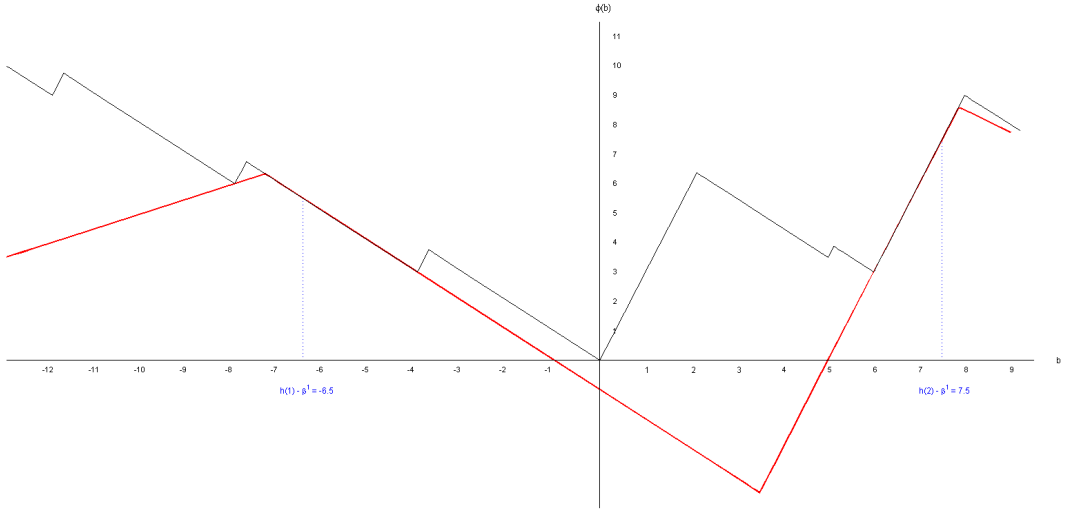
$$\begin{aligned} \min f(x) = \min \; & -3x_1 - 4x_2 + \sum_{s=1}^{2} 0.5Q(x,s) \\ & \text{s.t. } x_1 + x_2 \leq 5 \\ & x \in \mathbb{Z}_+ \end{aligned} \tag{11}$$

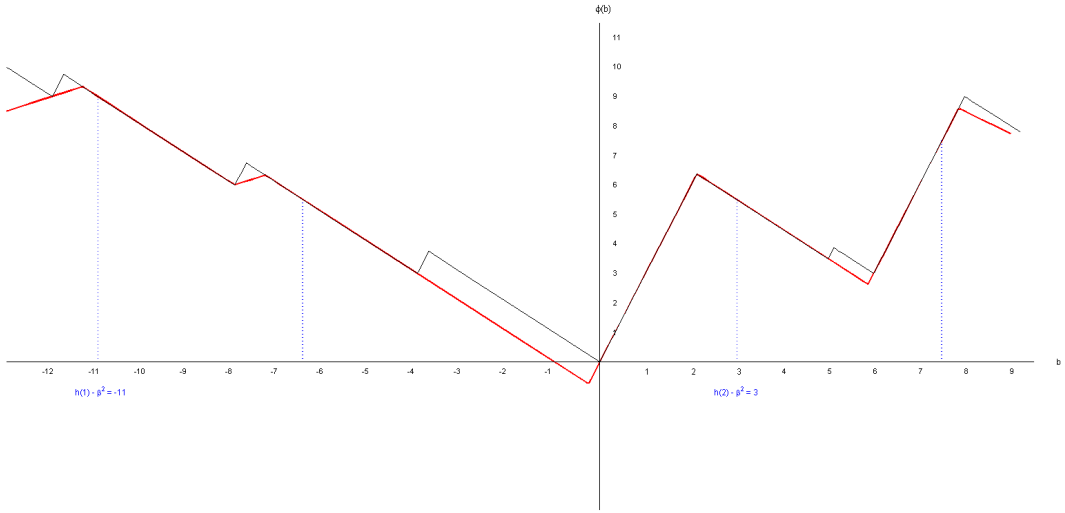
where

$$\begin{aligned} Q(x,s) = \min \; & 3y_1 + \frac{7}{2}y_2 + 3y_3 + 6y_4 + 7y_5 \\ & \text{s.t. } 6y_1 + 5y_2 - 4y_3 + 2y_4 - 7y_5 = h(s) - 2x_1 - \frac{1}{2}x_2 \\ & y_1, y_2, y_3 \in \mathbb{Z}_+, \; y_4, y_5 \in \mathbb{R}_+ \end{aligned} \tag{12}$$

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

Example





Solving the Master Problem

- A fundamental question is how to solver the master problem efficiently.
- It is a large integer program with many constraints, most of which will be redundant.
- It seems clear that we need to dynamically manage the LP relaxations.
- There are several options
  1. Put constraints in to a cut pool and add them only as they are violated.
  2. Add all constraints in the beginning, but aggressively remove slack ones.
- Our attempts have so far failed.
- Not having the full formulation makes branching less effective.
- We are currently investigating.

The Stochastic Server Location Problem sslp5-25 from the SIPLIB library has the following statistics. The second column indicates the number of variables in each second-stage problem. For each test problem, we denote the number of scenarios in parenthesis.

Problem	1st Stage			2nd Stage		
	All Vars	Int Vars	Const	All Vars	Int Vars	Const
sslp5-25	5	5	1	135	130	30

In all of the experiments preprocessing and heuristics are turned off. In the following tables, we use the following abbreviations
TLP: Average time spent in solving LPs (s) in all iterations
No. LPs: Average number of calls to LP solver in all iterations
No. Cuts: Average number of added cuts to LPs in all iterations
DT: Average depth of trees solved in all iterations
ST: Average size of trees solved in all iterations
and
DC: Dynamic cut generation SC: Static cut generation

	TLP		No. LPs		No. Cuts		ST		DT	
	DC	SC	DC	SC	DC	SC	DC	SC	DC	SC
iteration 1	0.008	0.004	2	1	10	0	1	1	0	0
iteration 2	0.024	0.016	10	5	21	3	7	3	3	1
iteration 3	0.168	0.06	49	21	45	6	19	15	5	4
iteration 4	0.25	0.15	58	38	95	54	33	21	6	5
iteration 5	0.6	0.22	128	43	235	75	61	33	13	7
iteration 6	0.72	0.3	147	41	177	101	63	31	13	6
iteration 7	0.8	0.38	118	40	255	110	61	27	11	6
iteration 8	1.38	0.47	186	53	347	145	119	29	16	5
iteration 9	0.86	0.94	114	75	313	192	71	23	15	10
iteration 10	0.94	0.96	118	71	331	127	63	51	13	9
iteration 11	1.48	1.57	164	125	451	338	75	43	12	10
iteration 12	2.61	1.35	2.77	102	551	233	137	69	15	9
iteration 13	3.82	1.60	376	109	663	240	191	51	14	10
iteration 14	4.37	2.09	435	134	748	477	203	59	14	10
iteration 15	3.94	2.15	393	115	703	315	139	61	14	10
iteration 16	6.98	2.7	571	136	918	639	255	81	16	10
iteration 17	14.42	2.73	1161	151	2289	377	447	79	17	3
Average	2.55	1.04	237.22	74.11	479.52	201.88	114.41	39.82	11.58	6.76

Table : sslp5-25(10)

Total running time is 22.2 seconds with the cuts vs 54.0 seconds with dynamic cuts.

	TLP		No. LPs		No. Cuts		ST		DT	
	DC	SC	DC	SC	DC	SC	DC	SC	DC	SC
sslp-5-25 (10)	2.55	1.04	237.22	74.11	479.52	201.88	114.41	39.82	11.58	6.76
sslp-5-25(15)	676.11	119.94	24.94	15.35	2557.58	370	977.47	153.11	32.38	3.50
sslp5-25(25)	94.17	9.96	44.23	22.23	2367.35	220.76	2554.82	265.23	6792.47	717.76

Table : Comparison of average statistics

	DC	SC
sslp-5-25 (10)	54	22.2
sslp-5-25(15)	622	81
sslp5-25(25)	1885	204

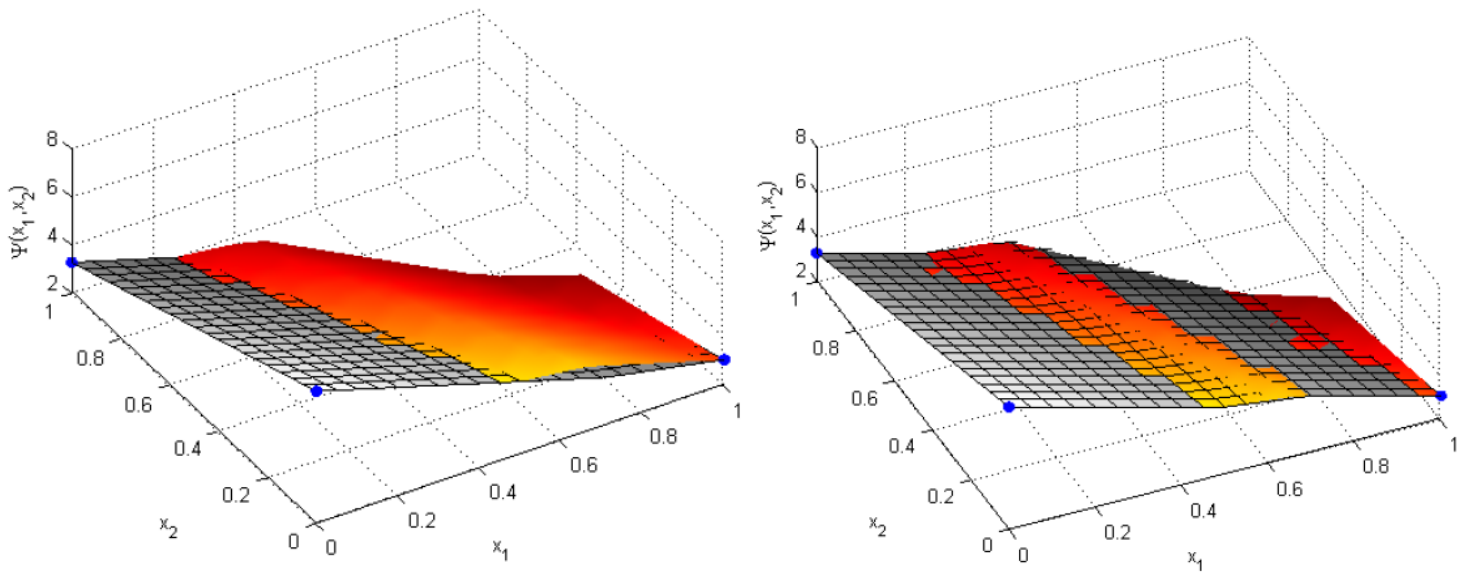
Table : Comparison of running times (seconds)

	TLP	No. LPs	No. Cuts	ST	DT	Running Time (s)
sslp-5-25 (50)	18.65	582.17	530.05	218.29	26.47	402
sslp-5-25(80)	29.21	783.52	538	198.64	33.17	630
sslp5-25(100)	99.25	1553	782	337	44	936

Table : Average of statistics for larger instances with static cut generation

Outline

Conclusions



- We have developed an algorithm for the two-stage problem with general mixed integer variables 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.
- There are still many questions to be answered about how to make all of this as efficient as possible.

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 can create an a priori cut pool.
- We so far have a fairly “naive” implementation and anticipate much improvement is possible.
- There is still much more to be learned about how to manage these piecewise polyhedral functions effectively.

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.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.

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.

A Hassanzadeh, T K Ralphs, and M Güzelsoy. On the value function of a mixed integer linear optimization problem and an algorithm for construction. Technical report, COR@L Laboratory, Lehigh University, 2014.

References II

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.

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.

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

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.

References III

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.

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.

References IV

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

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.