On Generalized Branching Methods for Mixed Integer Programming

Sanjay Mehrotra

Department of IE/MS

Northwestern University

Evanston, IL 60208

Joint Work with

Zhifeng Li

and

Huan-yuan Sheng

Organization

- Introduction and Review
- Adjoint Lattice
- Theoretical Results
- Computational Results
- Conclusions

Problem Formulation

$$\textbf{(GMICP)} \quad \begin{array}{rl} \text{minimize} & c_0(x) \\ \text{subject to} & Rx = r \\ & c_i(x) \le 0 \ \text{ for } i = 1, \ldots, l, \\ & x_z \in \mathbb{Z}^n, \ x_c \in \mathbb{R}^{\bar{n}}, \end{array}$$

where $x = (x_z, x_c)$, $c_i(x) : \mathbb{R}^{n+\bar{n}} \to \mathbb{R}$ for $i = 0, \ldots, l$ are convex functions. Note that $c_i(x) \le 0$ may include semi-definite and second-order cone constraints.

$$\textbf{(FILP)} \quad \text{Find } \{x \in \mathbb{Z}^n_+ \mid Ax = a\}.$$

Approaches to Branching

- Branching Strategies
  - Branching on variables
  - Most infeasible branching
  - Strong branching
  - Nested Cluster Branching
  - Pseudo-cost branching
  - Branching on constraints
- Branching on variables may not be efficient for solving general MIP.

An Example

$$
\begin{aligned}
\min \quad & x + y \\
s.t. \quad & px - qy \leq \frac{1}{2} \\
& -px + qy \leq \frac{1}{2} \\
& -p + \tfrac{1}{2} \leq y \leq \frac{1}{2}
\end{aligned}
$$

Relaxed optimal solution $(-q + \frac{q-1}{2p}, -p + \frac{1}{2})$. The optimal integer solution $(0, 0)$. BB-algorithm is $O(p)$. The input size is $O(\log p)$. The BB-method is exponential.

Matter of Definitions

Given $B = [b_1, \ldots, b_k] \in \mathbb{Z}^{n\times k}$, a *lattice* $\mathcal{L}$ is

$$\mathcal{L} \equiv \{x | x = \sum_{i=1}^{k} \lambda_i b_i, \lambda_i \in \mathbb{Z}\}.$$

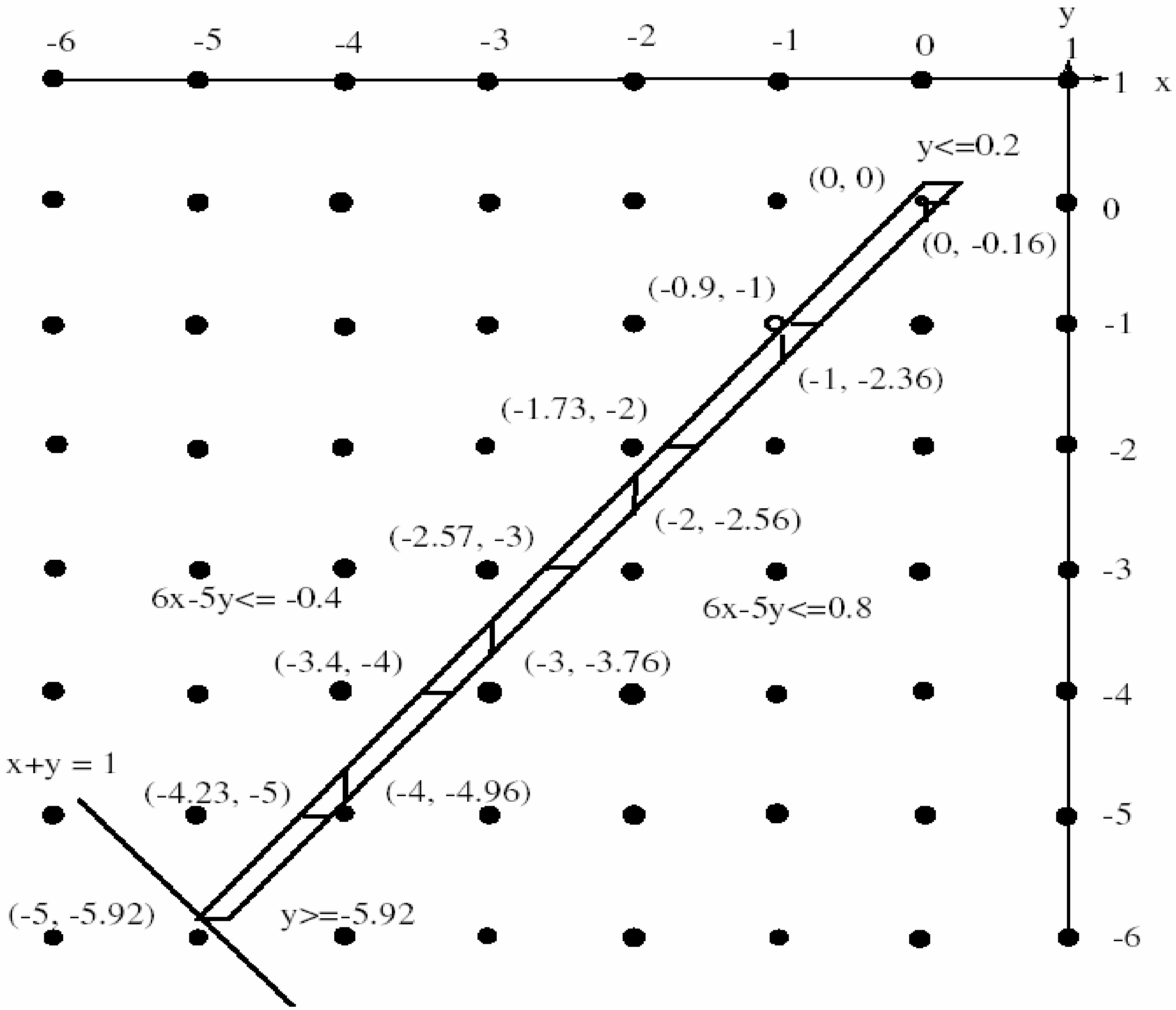
The set of vectors

$$\Lambda \equiv \{z \in \mathbb{Z}^n | Az = 0\}$$

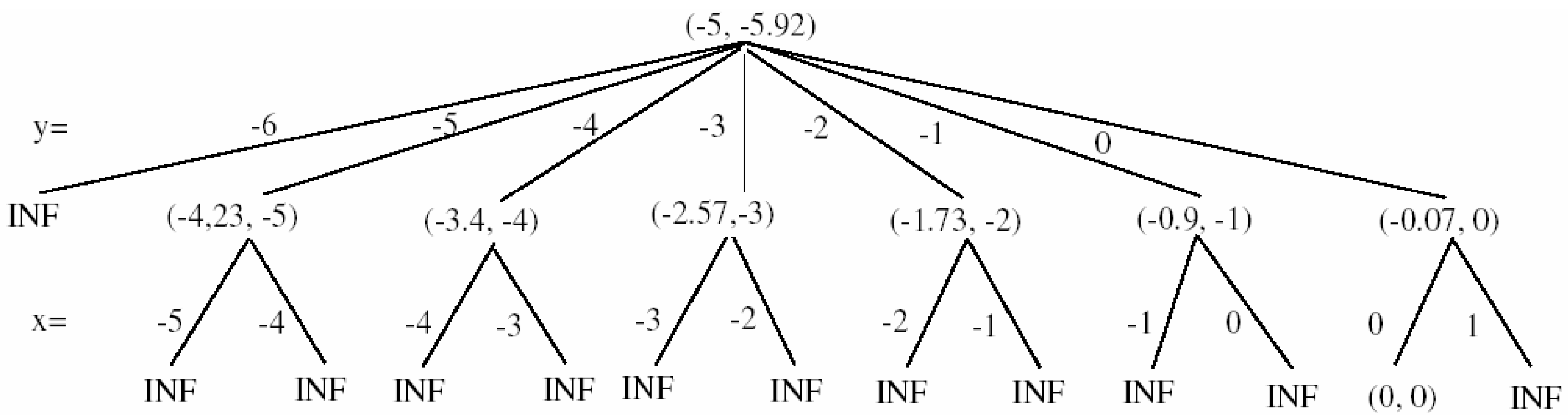
give a lattice called the integer kernel lattice of A, and a basis for this lattice is represented by Z. The *dual lattice* of $\mathcal{L}$:

$$\mathcal{L}^{\perp} \equiv \{z \in \mathbb{Q}^n | z^T x \in \mathbb{Z}, \text{ for all } x \in \mathcal{L}\}.$$

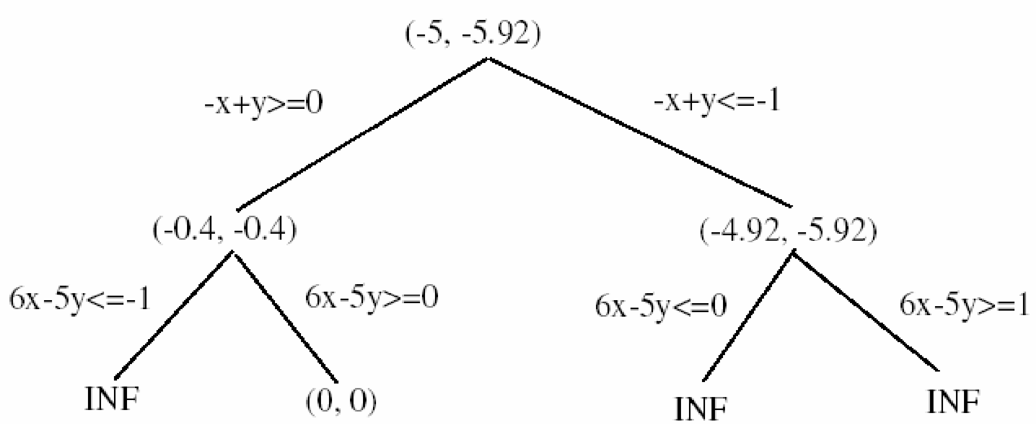
Example (continued)



An Example (continued)



The enumeration tree with branch-and-bound



The enumeration tree with branching on genearl hyperplanes

Lenstra's Algorithm

Step 0 **Dimension Reduction**

Transform $\mathcal{P} \equiv \{x \in \mathbb{R}^n_+ | Ax = a\}$ to $\mathcal{Y} \equiv \{y \in \mathbb{R}^k | Zy \geq -v\}$, ($x = Zy + v$, $Av = a$, Z is a Kernel Lattice of A, i.e., $AZ = 0$)

Repeat by Processing a node in the Branch and Bound Tree as

Step 1 **Ellipsoidal Rounding**

Find a positive definite matrix $Q \in \mathbb{R}^{k\times k}$ so that $\mathcal{E}(w, Q) \subseteq \mathcal{Y} \subseteq \mathcal{E}(w, Q/\gamma) \equiv \{y | \|y - w\|_Q \leq \gamma\}$.

Lenstra's Algorithm (continued)

Step 3 **Feasibility Check And Finding A Thin Direction**

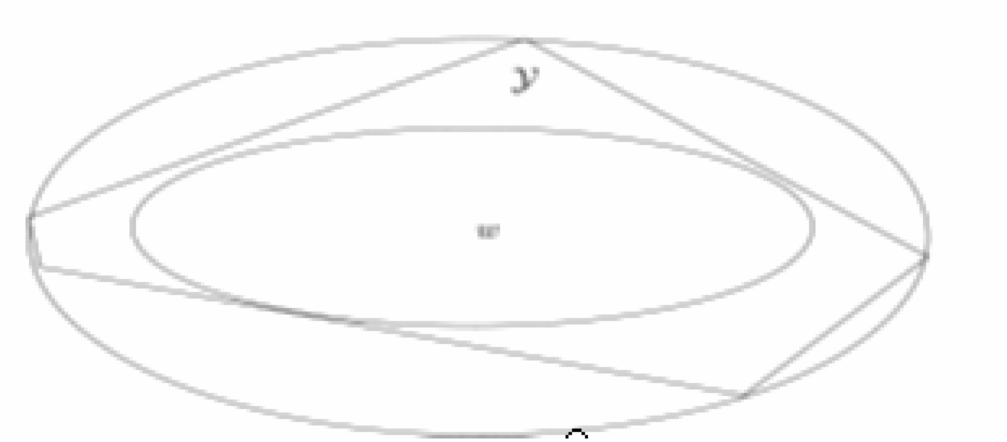
Find a LLL reduced basis $b_1, \ldots, b_k$ of $\mathbb{Z}^k$ such that b_i are short and nearly orthogonal under $\|\cdot\|_Q$. Round w to $y \in \mathbb{Z}^k$. If y is feasible, done; Otherwise, find a u satisfying $u^T b_1 = 0, \ldots, u^T b_{k-1} = 0$.

Step 4 **Branching On Hyperplanes**

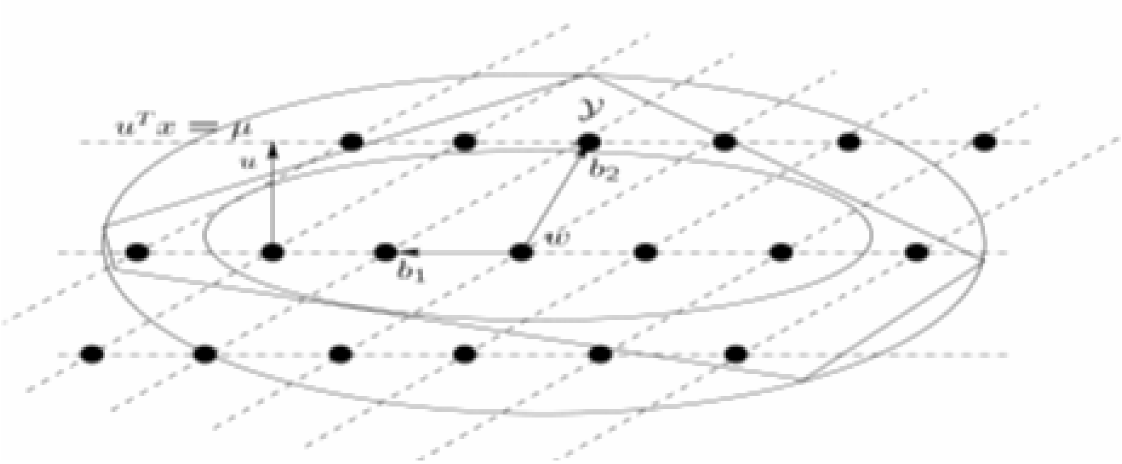
Add hyperplanes $u^T x = \mu$ for the entire range of μ, reduce problem dimensionality and Pick up a new node and return to Step 1.

Lenstra's Algorithm (Geometry)

Step 2:



Step 3:



Pervious Work

- Lenstra [83] proposed algorithm branching on hyperplane
- Grötschel, Lovász and Schrijver [84] used ellipsoidal approximation directly.
- Lovász and Scarf [92] developed Generalized Basis Reduction (GBR) algorithm that does not require ellipsoidal approximation and works with the original model, however, assumes full dimensionality.
- Cook, et. al. [93] implemented GBR algorithm for some hard network design problems.
- Wang [97] implemented GBR algorithm for LP and NLP (<100 integer variables).
- Aardal, Hurkens, and Lenstra [98,00], Aardal et. al. [00], Aardal and Lenstra [02], proposed a reformulation technique and solved some hard equality constrained integer knapsack and market split problems using this reformulation.
- Owen and Mehrotra [02] computationally showed that good branching disjunctions are available using heuristics for smaller problems in the MIPLIB testset.
- Gao and Zhang [02] implemented Lenstra's algorithm and tested an interior point algorithm for finding the maximum volume ellipsoid.

Issues

- Dimension reduction… hmm…
  - Don’t know what to do when continuous variables are present
    - Continuous variables are “projected” out
  - How to do this for the more general convex problem?
  - It is just nice to work in the original space!
- Basis Reduction at every node…
  - Finding a feasible solution or
  - finding good branching directions
- Ellipsoidal Approximation
  - We should use the modern technologies!
- Feasibility problem vs. Optimization Problem
  - Disjunctive branching (instead of branching on hyperplanes)

Adjoint Lattice

The lattice generated by an integer matrix Z^* satisfying $Z^T Z^* = I$ is called an *adjoint lattice* of A (associated with Z). An adjoint lattice exists and in fact, it is not unique.

Example 1 (Example 2.2 in Aardal et al. [02])

$$\mathcal{P} := \{x : \ 2x_1 + 4x_2 + 5x_3 = 8, 0 \le x_j \le 1, 1 \le j \le 3\}$$

i.e., $A = [2, 4, 5]$, $a = 8$, and $v = [0, 2, 0]^T$ is a particular solution of $Ax = a$. A kernel lattice basis is given by $Z = \begin{bmatrix} -2 & 5 \\ 1 & 0 \\ 0 & -2 \end{bmatrix}$ and $Z^* = \begin{bmatrix} 0 & 1 \\ 1 & 2 \\ 0 & 2 \end{bmatrix}$, respectively. Note that $Z^* + A^T[0, -1] = \begin{bmatrix} 0 & -1 \\ 1 & -2 \\ 0 & -3 \end{bmatrix}$ gives another adjoint lattice basis.

Branching Hyperplane finding Problem

Proposition: In the branching hyperplane finding problem for $\mathcal{Y}$:

$$\min_{p \in \mathbb{Z}^k \setminus 0} \left\{ \max_{y \in \mathcal{Y}} p^T y - \min_{y \in \mathcal{Y}} p^T y \right\},$$

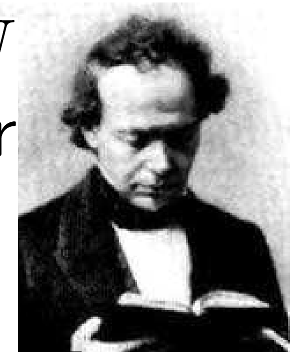
equals

$$\min_{u \in Z^* \setminus 0} \left\{ \max_{x \in \mathcal{P}} u^T x - \min_{x \in \mathcal{P}} u^T x \right\},$$

$\mathcal{P} \equiv \{x | Ax = a, x \geq 0\}$

Find An integral Adjoint Basis

This goes back to the gcd calculation. Let U be such that $AU = [H : 0]$, where H is a lower triangular matrix. Let $U = [U_I : Z]$, then



Hermite

$$AU_I = H, AZ = 0.$$

Let $U^{-1} = \begin{bmatrix} U_A^T \\ Z^{*T} \end{bmatrix}$. Then Z^* is a basis of Λ^* because

$$Z^{*T} Z = Z^T Z^* = I.$$

By Λ^* denote the lattice generated from the columns of Z^*.

Ellipsoidal Approximation

- John [48] showed the existence of k-rounding for a given convex set with non-empty interior point.
- Lenstra [83] gave a constructive procedure for finding an ellipsoidal rounding.
- Interior-Point Methods: Analytic center, Vaidya Center, Volumetric center and corresponding ellipsoidal rounding.

Analytic Center:

$$\max \rho(x) \equiv \sum_{i=1}^{n} \ln x_i \text{ satifying } x \in \mathcal{P}.$$

$$\mathcal{E}(w, Q) \subseteq \mathcal{P} \subseteq \mathcal{E}(w, Q/\gamma),$$

where $Q = \nabla^2 \rho(w)$.

The Quality of Approximation

$\gamma = O(n)$ in the linear case. When general convex constraints are present, if the feasible set admits a self-concordant barrier, $\gamma = O(\theta)$, where θ is the complexity value of a self-concordant barrier. Note also that the log-barrier is well defined over twice continuously differentiable convex functions.

Bound on Minimum Width Using Ellipsoidal Approximation

Theorem 1 Either there exists a branching direction $u \in \Lambda^*$ such that

$$\mathcal{W}(u, \mathcal{P}) \leq 2\gamma(\frac{3}{\sqrt{2}})^k,$$

or we can find a feasible solution of (FILP) (or Feasibility integer convex program). We can compute this direction in polynomial time by using basis reduction algorithm of Lenstra, Lenstra, and Lovasz (or other such algorithm).

How to Solve Minimum Width Problems

Definition (Lenstra, Lenstra, and Lovász Reduced Basis). A basis $b_1, \ldots, b_k$ of a lattice $\mathcal{L}$ is called LLL-reduced basis (under $\| \cdot \|_E$) for a given $\delta \in (.25, 1)$, if the following two conditions hold for $i = 1, \ldots, k-1$:

(1) $|\widehat{b}_{k+1}\|_E^2 \geq (\delta - \Gamma_{k,k+1}^2)\|\widehat{b}_k\|_E^2$, where $\|\widehat{b}_k\|$ are Gram-Schmidt orthogonal vectors under $\| \cdot \|_E$, and $\Gamma_{i,i+1}$ are the corresponding Gram-Schmidt coefficients.

(2) $|\Gamma_{j,i}| \leq 1/2$, for $1 \leq j < i \leq n$.

Lovasz and Scarf Basis

Definition (Lovász and Scarf Reduced Basis). A basis $b_1, \ldots, b_k$ of Λ^* is called LS-reduced basis for a given $0 < \epsilon < \frac{1}{2}$ if the following two conditions hold for $i = 1, \ldots, k-1$:

(1) $F_i(b_{i+1} + \mu b_i) \geq F_i(b_{i+1})$ for integral μ,
(2) $F_i(b_{i+1}) \geq (1-\epsilon) F_i(b_i)$,

where

$$F_i(u) = \max \; \{u^T x - u^T y : x \in \mathcal{X}, y \in \mathcal{X}, \; b_1^T(x-y) = 0, \ldots, b_{i-1}^T(x-y) = 0\}.$$

An interpretation of the Aardal, Hurkens and Lenstra Reformulation

Find a LLL-reduced Z in L_2 norm and then reduce the problem to a lower dimension. Branch on the coordinates $e_i, i = k, \ldots, 1$.

Result: Finding an LLL reduced Z is "equivalent" to finding an LLL reduced Z^* under $\|\cdot\|_{P_A}$, where $P_A = I - A^T(AA^T)^{-1}A$.

Mixed Integer Programming

- Feasibility Mixed Integer Linear Program (FMILP) is to

$$\text{Find } x \in \bar{\mathcal{X}},$$

$$\text{where } \bar{\mathcal{X}} \equiv \left\{ x = \begin{bmatrix} x_z \\ x_c \end{bmatrix} \,|Rx = r, x_z \in \mathbb{Z}_+^n, x_c \in \mathbb{R}_+^{\bar{n}} \right\},$$

$R = \begin{bmatrix} B & C \\ A & 0 \end{bmatrix}$, $r = \begin{bmatrix} b \\ a \end{bmatrix}$, $B \in \mathbb{Z}^{\bar{m} \times n}$, and $C \in \mathbb{Z}^{\bar{m} \times \bar{n}}$.

Mixed Integer Problem

Theorem 2: There exists a branching direction $u \in \Lambda^*$ such that

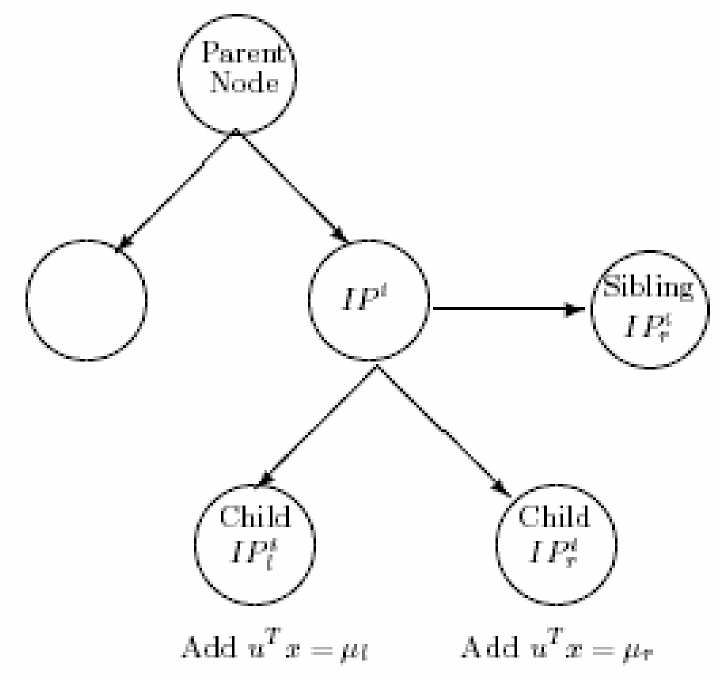
$$\mathcal{W}\left(\begin{bmatrix} u \\ 0 \end{bmatrix}, \bar{\mathcal{X}}\right) \leq \gamma(3/\sqrt{2})^k,$$

or we can find a feasible solution of the mixed integer convex feasible set. Note that γ is a function of $(n + \bar{n})$ instead of n.

Generalized B&B Method

IMPACT (Integer Mathematical Programming Advanced Computational Tool) implementation of GBB.

- A growing search tree



Knapsack Test Problems

Numerical Results on Hard Knapsack Problems

Program	LLL-L2		LLL-Ellip-R		LLL-Ellip-E		GBR-R		GBR-E		GBROnly-R		GBROnly-E		CPLEX
	#N	#T	#N	#T	#N	#T	#N	#T	#N	#T	#N	#T	#N	#T	#N
Law1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	50MN
Law2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2586612
Law3	3	1	3	1	3	1	3	1	3	1	3	1	3	1	50MN
Problem5	5	3	6	3	3	1	6	3	3	1	6	3	3	1	50MN
Problem6	3	1	3	1	5	3	3	1	3	1	3	1	3	1	50MN
Problem7	5	3	5	3	3	1	5	3	3	1	5	3	3	1	50MN
Problem8	3	1	3	1	3	1	3	1	3	1	3	1	3	1	50MN
Problem15	34	21	28	17	17	12	35	29	24	13	35	29	24	16	80,624
Problem16	46	31	21	14	15	10	30	23	15	9	30	23	15	9	484,564
Problem17	39	29	24	16	21	17	27	19	24	17	27	19	22	17	62,792
Problem18	58	32	29	23	11	7	29	22	17	12	29	22	15	11	110,711
Problem19	57	38	32	21	21	13	34	23	22	15	34	23	22	16	116,304
Problem20	27	18	21	18	14	11	20	17	13	11	20	17	13	11	230,562

Market Split Problems

	LLL-R(P_A)				LLL-R(P)	LLL-E(P)	GBR-R	GBR-E
Problem	#nodes	LLL(iter)	sec.	width	nodes	nodes	nodes	nodes
msp31	3	428	0.15	0	3	5	3	7
msp32	5	459	0.14	1	5	3	9	5
msp33	1	565	0.15	0	1	1	1	1
msp34	1	619	0.16	0	1	1	1	1
msp41	66	1321	0.73	2	66	117	100	315
msp42 F	21	1281	0.6	2	21	22	47	102
msp43	15	1728	0.57	1	15	15	15	15
msp44	19	1645	0.57	1	19	23	24	17
msp51 F	817	1895	8.32	3	817	2,498	†	†
msp52	438	2308	4.41	2	438	986	636	†
msp53	661	2341	5.78	3	661	1,059	899	†
msp54	214	3115	2.72	2	214	256	177	†
msp55	84	2929	1.84	2	84	256	†	†
msp61	4,258	3978	44.23	2	4,258	12,455	†	†
msp62	6,587	4693	77.39	3	6,587	12,030	†	†
msp63	13,725	3879	136.5	3	13,725	14,176	†	†
msp64	141,852	3012	1407	1	141,852		†	†
msp31Gen F	27	428	0.00	4	49	51	52	†
msp32Gen F	13	459	0.00	3	25	30	19	†
msp33Gen	13	565	0.01	2	13	11	28	†
msp34Gen	22	619	0.02	3	19	13	21	†
msp41Gen F	373	1,321	0.03	6	1,066	2,289	753	†
msp42Gen F	507	1,281	0.01	7	3,464	1,994	2,213	†
msp43Gen	1,817	1,728	0.16	5	774	763	935	†
msp44Gen	870	1,645	0.10	5	423	540	1,511	†
msp51Gen F	4,006	1,895	.25	10	42,057	15,579	10,929	†
msp52Gen F	54,994	2,308	2.87	9	428,826	156,181	137,216	†
msp53Gen F	10,755	2,341	.53	7	90,456	38,186	37,385	†
msp54Gen	79,941	3,115	4.59	6	67,476	32,926	55,515	†
msp55Gen	53,761	2,929	2.68	7	65,119	55,113	†	†

Legends: + indicates out-of-memory; † numerical failure due to LP optimality tolerances.

Larger Market Split Problems

				LLL-R(P)			
Problem	m	n	Status	#nodes	LLL(iter)	sec.	$\mathcal{W}_I(Z_1^*)$
msp41	4	30	N	130	1321	0.73	2
msp42	4	30	F	77	1281	0.6	2
msp43	4	30	N	34	1728	0.57	1
msp44	4	30	N	46	1645	0.57	1
msp51	5	40	F	2,395	1895	8.32	3
msp52	5	40	N	939	2308	4.41	2
msp53	5	40	N	1,393	2341	5.78	3
msp54	5	40	N	442	3115	2.72	2
msp55	5	40	N	196	2929	1.84	2
msp61	6	50	N	9,042	3978	44.23	2
msp62	6	50	N	15,999	4693	77.39	3
msp63	6	50	N	29,791	3879	136.5	3
msp64	6	50	N	316,383	3012	1407	1
msp71	7	60	F	10,512,209 **	3610	637,577	1
msp72	7	60	F	3,764,124 **	3463	68,067	1
msp73	7	60	N	171,466	6281	1,028	1
msp74	7	60	N	216,972	6356	1,249	4

of LLL iterations and GBB Tree Size

	LLL-R(P) ($\delta = 0.9$)			LLL-R(P) ($\delta = 0.99$)		
Problem	#nodes	#LLL(iter)	sec.	#nodes	#LLL(iter)	sec.
msp41	265	723	1.02	130	1321	0.73
msp42	301	659	0.99	77	1281	0.6
msp43	67	1038	0.58	34	1728	0.57
msp44	52	1116	0.55	46	1645	0.57
msp51	8,300	1071	23.42	2,395	1895	8.32
msp52	2,694	1397	10.22	939	2308	4.41
msp53	3,752	1270	14.09	1,393	2341	5.78
msp54	746	1703	3.54	442	3115	2.72
msp55	1,987	1518	4.81	196	2929	1.84
msp61	38,611	2162	172.5	9,042	3978	44.23
msp62	29,524	2412	144.7	15,999	4693	77.39
msp63	106,192	2111	469.4	29,791	3879	136.5
msp64	1,203,447	1537	5523	316,383	3012	1407

Conclusions

- Developed general branching methods for Linear and Convex Mixed Integer Programs.
- It is possible to develop stable codes for Dense Difficult Market Split problems using Adjoint Lattice Basis.