On the Complexity and Solution of Inverse Mixed Integer Linear Programs

Aykut Bulut[1]

Joint work with:
Ted Ralphs[1]

[1]COR@L Lab, Department of Industrial and Systems Engineering, Lehigh University

INFORMS Annual Meeting
October 2013

Outline

1. Inverse MILP
2. Complexity
3. Algorithm
4. Computational Results

Outline

1. Inverse MILP
2. Complexity
3. Algorithm
4. Computational Results

Outline

1. Inverse MILP
2. Complexity
3. Algorithm
4. Computational Results

Outline

1. Inverse MILP
2. Complexity
3. Algorithm
4. Computational Results

Definitions

For a given $d \in \mathbb{R}^n$ and $\mathcal{P}$, we consider a MILP

$$z_{IP} = \min_{x \in \mathcal{P}} d^T x, \tag{1}$$

where,

$$\mathcal{P} = \{x \in \mathbb{R}^n | Ax = b, x \geq 0\} \cap (\mathbb{Z}^r \times \mathbb{R}^{n-r}).$$

For a given $c \in \mathbb{R}^n$, $x^0 \in \mathcal{P}$, the inverse problem is defined as follows.

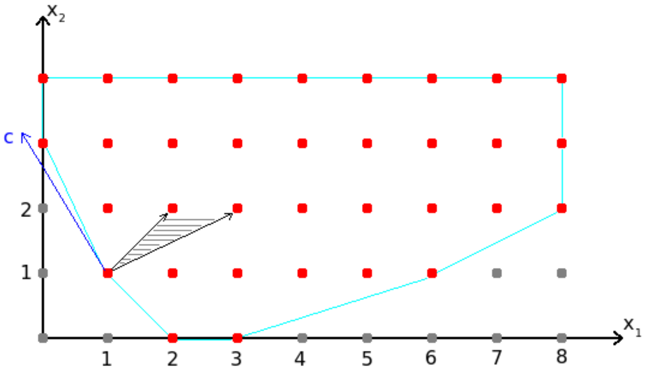
$$\begin{aligned} & \min \|c - d\| \\ & s.t. \\ & d^T x^0 \leq d^T x \qquad \forall x \in \mathcal{P}. \end{aligned} \tag{2}$$

Assumption: $\mathcal{P}$ is bounded.

A Small Example

Define forward problem feasible set $\mathcal{P}$ as follows and let $c = (-3, 5)$, $x^0 = (1, 1)$.

Figure: $\mathcal{P}$, its convex hull and feasible d cone



Definitions

$$
\begin{aligned}
& \min \|c - d\| \\
& s.t. \\
& d^T x^0 \le d^T x \qquad \forall x \in \mathcal{P}.
\end{aligned} \tag{4}
$$

- Model can be linearized for l_1 and l_∞ norms.
- Convex hull of $\mathcal{P}$ is a polytope.
- Last constraint set can be represented with the set of extreme points of convex hull of $\mathcal{P}$.
- Let $\mathcal{E}$ be the set of extreme points of convex hull of $\mathcal{P}$, $\mathcal{E}$ is finite.

Inverse MILP with l_1 norm

$$
\begin{aligned}
z_{IP}^1 &= \min \sum_{i=1}^{n} \theta_i \\
s.t. \\
c_i - d_i &\leq \theta_i && \forall i \in \{1, 2, \ldots, n\} \\
d_i - c_i &\leq \theta_i && \forall i \in \{1, 2, \ldots, n\} \\
d^T x^0 &\leq d^T x && \forall x \in \mathcal{E}.
\end{aligned}
\tag{3}
$$

Inverse MILP with l_∞ norm

$$
\begin{aligned}
z_{IP}^{\infty} &= \min y \\
& s.t. \\
& c_i - d_i \le y && \forall i \in \{1, 2, \ldots, n\} \\
& d_i - c_i \le y && \forall i \in \{1, 2, \ldots, n\} \\
& d^T x^0 \le d^T x && \forall x \in \mathcal{E}.
\end{aligned}
\tag{4}
$$

For the remainder of the presentation, we deal with the case of l_∞ norm. Let $\mathcal{S}$ represent feasible set of the inverse IP, defined as

$$\mathcal{S} = \left\{ (y, d) \in \mathbb{R} \times \mathbb{R}^n | y \ge \|c - d\|_\infty, d^T (x^0 - x) \le 0 \; \forall x \in \mathcal{E} \right\}.$$

Note that $\mathcal{S}$ is a polyhedron.

Polynomially Solvable Forward Problems

Ahuja and Orlin [1] determines the complexity of inverse problem when the forward problem is polynomially solvable.

Theorem

(Ahuja and Orlin [1]) If a forward problem is polynomially solvable for each linear cost function, then corresponding inverse problems under l_1 and l_∞ are polynomially solvable.

Forward Problems

Define the following problems related to MILP and inverse MILP.

Definition

MILP decision problem: Given $\gamma \in \mathbb{Q}$ decide whether $d^T x \leq \gamma$ holds for some $x \in \mathcal{P}$.

Definition

MILP optimization problem: Find solution vector x^* such that $x^* \in \text{argmin}_{x \in \mathcal{P}} d^T x$ or decide the problem is unbounded or decide the problem is infeasible.

Inverse Problems

Definition

Inverse MILP decision problem: Given $\gamma \in \mathbb{Q}$ decide whether $y \leq \gamma$ holds for some $(y, d) \in \mathcal{S}$.

Definition

Inverse MILP optimization problem: Find solution vector (y^*, d^*), such that $(y^*, d^*) \in \text{argmin}_{(y,d)\in\mathcal{S}} y$.

Definition

Inverse MILP separation problem: Given a vector $(\bar{y}, \bar{d}) \in \mathbb{Q} \times \mathbb{Q}^n$, decide whether $(\bar{y}, \bar{d})$ is in $\mathcal{S}$, and if not, find a hyperplane that separates $(\bar{y}, \bar{d})$ from $\mathcal{S}$, i.e., find $\pi \in \mathbb{Q}^{n+1}$ such that $\pi^T \begin{bmatrix} \bar{y} \\ \bar{d} \end{bmatrix} < \min \left\{ \pi^T \begin{bmatrix} y \\ d \end{bmatrix} | (y, d) \in \mathcal{S} \right\}$.

Component 1–An Observation

Recall inverse MILP optimization problem.

$$
\begin{aligned}
z_{IP}^{\infty} &= \min y \\
s.t. & \\
c_i - d_i &\leq y && \forall i \in \{1, 2, \ldots, n\} \\
d_i - c_i &\leq y && \forall i \in \{1, 2, \ldots, n\} \\
d^T x^0 &\leq d^T x && \forall x \in \mathcal{E}.
\end{aligned}
$$

Definition

Inverse MILP separation problem: Given a vector $(\bar{y}, \bar{d}) \in \mathbb{Q} \times \mathbb{Q}^n$, decide whether $(\bar{y}, \bar{d})$ is in $\mathcal{S}$, and if not, find a hyperplane that separates $(\bar{y}, \bar{d})$ from $\mathcal{S}$, i.e., find $\pi \in \mathbb{Q}^{n+1}$ such that $\pi^T \begin{bmatrix} \bar{y} \\ \bar{d} \end{bmatrix} < \min \left\{ \pi^T \begin{bmatrix} y \\ d \end{bmatrix} \middle| (y, d) \in \mathcal{S} \right\}$.

Component 2–GLS Theorem

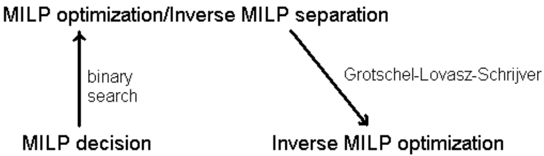
The following theorem by Grötschel et al. indicates the relationship between separation and optimization problems.

Theorem

(Grötschel et al. [2]) Given an oracle for the separation problem, the optimization problem over a given polyhedron with linear objective can be solved in time, polynomial in φ, n and the encoding length of objective coefficient vector, where facet complexity of polyhedron is at most φ.

Complexity of MILP optimization/decision problems

Figure: Problem relations



Theorem

Inverse MILP optimization problem under l_∞/l_1 norm is solvable in time polynomial of φ, $n + 1/2n$, and encoding length of $(1, 0, \ldots, 0)/(1, \ldots, 1, 0, \ldots, 0)$, given an oracle for the MILP decision problem.

Corollary

Inverse MILP decision problem is in Δ_2^{P}.

Inverse MILP decision is Δ_2^{P}–complete

Inverse Decision problem is Δ_2^{P}–complete by a reduction from DSAT problem.

Definition

Deterministic formula: Let $Y_1, Y_2, \ldots, Y_k, \{x_1, x_2, \ldots, x_{k-1}\}$ be sets of Boolean variables. A formula F involving these variables is said to be *deterministic* if it consists of the conjuction of the following clauses.

(a) For each $y \in Y_1 \cup Y_k$ either (y) or $(\bar{y})$ is a clase of F.

(b) For each $j = 1, 2, \ldots, k-1$ and each $y \in Y_{j+1}$, there are two sets of conjuctions of literals from $Y_j \cup \{x_j\}$, say $\{C_i\}$ and $\{\mathsf{D}_i\}$, such that $(C_i \rightarrow y)$ and $(\mathsf{D}_i \rightarrow \bar{y})$ are all clauses of F, and, furthermore, for any truth assignment for $Y_j \cup \{x_j\}$ exactly one of the conjuctions in $\{C_i\} \cup \{D_i\}$ is true.

Definition

Deterministic Satisfiability (DSAT) is the following computational problem. Given k formulas $F_0(x_1, \ldots, x_{k-1}, Y_1, \ldots, Y_k), F_1(Y_1, Z_1), \ldots, F_{k-1}(Y_{k-1}, Z_{k-1})$, where $\{x_1, \ldots, x_{k-1}\}, Y_1, \ldots, Y_k, Z_1, \ldots, Z_{k-1}$ are disjoint sets of variables and F_0 is a deterministic formula, is there a truth assignment $\hat{x}_1, \ldots, \hat{x}_{k-1}, \hat{Y}_1, \ldots, \hat{Y}_k$ for x and y variables such that

(a) F_0 is satisfied, and

(b) $F_j(\hat{Y}_j, Z_j))$ is satisfiable iff $\hat{x}_j = true$.

Theorem

(Papadimitriou [3]) DSAT is Δ_2^{P}–complete.

Proof follows by reducing a Deterministic Truing Machine with an NP oracle to DSAT.

Inverse MILP decision is Δ_2^{P}–complete

- DSAT is canonical Δ_2^{P}–complete problem, like SAT for NP–complete.
- Papadimitriou defines DSAT to reduce it to unique optimum problem and shows that it is Δ_2^{P}–complete [3].
- Using similar techniques DSAT can be reduced to inverse TSP decision version.
- Inverse TSP decision version is Δ_2^{P}–complete. Since it is a special case of inverse MILP decision, inverse MILP decision is also Δ_2^{P}–complete.

Theorem

Inverse MILP decision problem is Δ_2^{P}–complete.

Proposed Algorithm

First, we define two parametric problems named P_k and $InvP_k$ as follows

$$\min_{x \in \mathcal{P}} d^{kT} x \tag{P_k}$$

$$\begin{aligned}
& \min y \\
& s.t. \\
& c_i - d_i \leq y && \forall i \in \{1, 2, \dots, n\} \\
& d_i - c_i \leq y && \forall i \in \{1, 2, \dots, n\} \\
& d^T x^0 \leq d^T x && \forall \mathcal{E}^{k-1}
\end{aligned} \tag{$InvP_k$}$$

where $\mathcal{E}^{k-1}$ is the set of extreme points found so far.

Proposed Algorithm

Algorithm 1

$k \leftarrow 1$
$d^k \leftarrow c$
Solve P_k, $x^k \leftarrow x^*$
while $d^{kT}(x^0 - x^k) > 0$ **do**
  $k \leftarrow k + 1$
  Solve $InvP_k$, $d^k \leftarrow d^*$
  Solve P_k, $x^k \leftarrow x^*$
end while

- x^k is an optimal solution to P_k, whereas d^k is an optimal solution to $invP_k$.
- The algorithm stops when a generated cut is not violated by the current solution.

Table: MIPLIB Iteration number–l_1

p. name	pert. 0.1	pert. 0.2	pert. 0.3	pert. 0.4	pert. 0.5	pert. 0.6
bell3a	1	1	2	2	11	7
blend2	2	2	2	2	2	2
dcmulti	1	5	1	27	21	47
egout	20	20	20	20	20	20
enigma	1	1	1	1	1	1
flugpl	1	1	1	1	1	1
gen	1	142	188	852	844	1578
gt2	1	1	1	1	1	1
l152lav	16	23	46	46	46	46
lseu	1	1	1	1	1	1
mas74	1	1	1	1	1	1
misc03	1	1	1	1	1	1
misc06	1	1	1	1	1	1
mod008	3	3	6	6	6	6
mod010	51	67	311	390	390	390
modglob	1	1	1	1	1	1
p0201	1	1	4	4	4	4
p0282	1	1	1	1	1	1
pk1	1	1	1	1	1	1
pp08aCUTS	11	12	4	4	14	3
qiu	1	16	10	10	10	10
rgn	1	1	1	1	1	1
rout	1	1	1	1	1	1
stein27	1	1	1	1	1	1
vpm1	1	1	1	1	1	1

Table: MIPLIB Iteration number–l_{inf}

p. name	pert. 0.1	pert. 0.2	pert. 0.3	pert. 0.4	pert. 0.5	pert. 0.6
air03	2	3	2	2	2	2
blend2	3	3	3	2	3	2
cap6000	6	27	30	30	30	32
dcmulti	1	3	1	10	17	21
egout	8	8	8	9	9	9
enigma	1	1	1	1	1	1
flugpl	1	1	1	1	1	1
gen	1	21	35	37	220	179
gt2	1	1	1	1	1	1
harp2	6	6	6	6	7	8
l152lav	4	3	4	4	4	4
lseu	1	1	1	1	1	1
mas74	1	1	1	1	1	1
misc03	1	1	1	1	1	1
misc06	1	1	1	1	1	1
mod008	2	2	3	3	3	2
mod010	4	3	3	3	3	3
modglob	1	1	1	1	1	1
p0201	1	1	2	3	3	3
p0282	1	1	1	1	1	1
pk1	1	1	1	1	1	1
pp08aCUTS	11	11	5	5	8	5
qnet1	1	1	1	1	1	1
rgn	1	1	1	1	1	1
rout	1	1	1	1	1	1
stein27	1	1	1	1	1	1
vpm1	1	1	1	1	1	1

Table: TSPLIB Iteration number–l_1

p. name	pert. 0.1	pert. 0.2	pert. 0.3	pert. 0.4	pert. 0.5	pert. 0.6
att48	1	1	3	4	9	24
berlin52	1	1	1	1	1	1
bier127	73	103	186	307	298	358
burma14	1	1	1	1	1	1
ch130	1	14	31	52	88	81
ch150	1	7	15	20	49	49
eil101	1	4	7	4	8	6
eil51	1	2	4	4	7	16
eil76	1	3	2	4	4	6
kroA100	3	3	6	10	17	38
kroC100	1	3	4	4	8	14
kroD100	1	4	13	24	43	138
kroE100	1	13	13	17	27	26
lin105	1	2	6	18	20	20
pr107	2	13	23	30	38	39
pr124	2	3	6	8	15	13
pr152	3	41	18	18	24	45
rat99	1	10	4	13	21	22
rd100	1	3	8	8	11	11
st70	1	4	13	14	11	14
u159	1	1	1	8	9	13
ulysses16	1	1	1	1	1	1
ulysses22	1	1	3	3	3	5

Table: TSPLIB Iteration number–l_{inf}

p. name	pert. 0.1	pert. 0.2	pert. 0.3	pert. 0.4	pert. 0.5	pert. 0.6
att48	1	1	2	2	3	5
berlin52	1	1	1	1	1	1
bier127	6	5	6	5	6	6
burma14	1	1	1	1	1	1
ch130	1	5	4	6	6	6
ch150	1	4	4	5	5	5
eil101	1	2	2	2	2	2
eil51	1	2	2	2	2	3
eil76	1	3	4	3	3	2
kroA100	2	2	2	3	3	3
kroA150	4	3	5	5	5	5
kroB200	3	4	3	5	6	6
kroC100	1	3	3	3	2	3
kroD100	1	2	4	4	4	5
kroE100	1	5	5	5	5	4
lin105	1	1	3	2	4	4
pr107	1	4	4	6	4	7
pr124	2	3	2	3	3	3
pr136	2	3	4	6	6	6
pr144	3	4	5	5	6	4
pr152	2	3	5	5	3	3
pr76	1	2	3	3	3	3
rat99	1	3	2	4	4	4
rd100	1	2	3	3	3	3
st70	1	2	4	5	4	5
u159	1	1	2	3	3	4
ulysses16	1	1	1	1	1	1
ulysses22	1	1	3	3	2	5

References

Ravindra K. Ahuja and James B. Orlin.
Inverse optimization.
Operations Research, 49(5):771–783, September/October 2001.

Martin Grötschel, Lászlo Lovász, and Alexander Schrijver.
Geometric Algorithms and Combinatorial Optimization, volume 2 of *Algorithms and Combinatorics*.
Springer, second corrected edition edition, 1993.

Christos H. Papadimitriou.
On the complexity of unique solutions.
J. ACM, 31(2):392–400, March 1984.

End of presentation

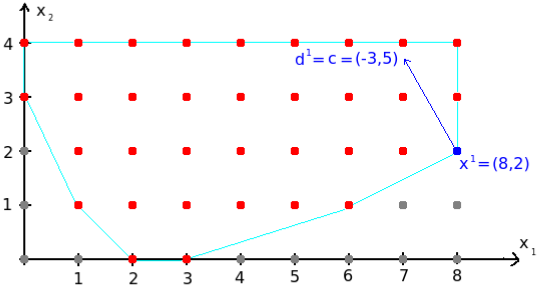
This is end of presentation!

Thank you for listening!

A Small Example: Iteration 1

Let $\mathcal{P}$ and convex hull of $\mathcal{P}$ given as in Figure. Let $c = (-3, 5)$ and $x^0 = (1, 1)$. We know $d^1 = c$. When objective coefficient of forward problem is d^1, optimal solution is $x^1 = (8, 2)$.

Figure: Iteration 1, d^1 and x^1

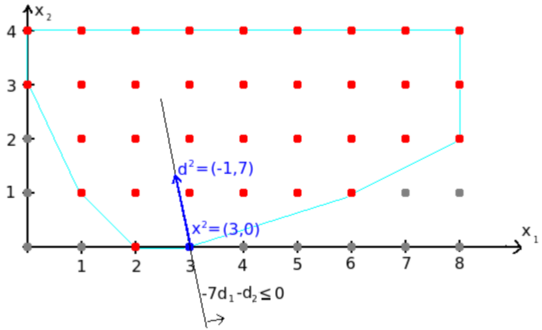


Using x^1, we generate cut $d^\top(x^0 - x^1) \leq 0$, i.e. $-7d_1 - d_2 \leq 0$.

A Small Example: Iteration 2

Following figure shows feasible cone for d. d^2 is the inverse optimal with the current cut. x^2 is the forward optimal solution when d^2 is objective coefficient vector.

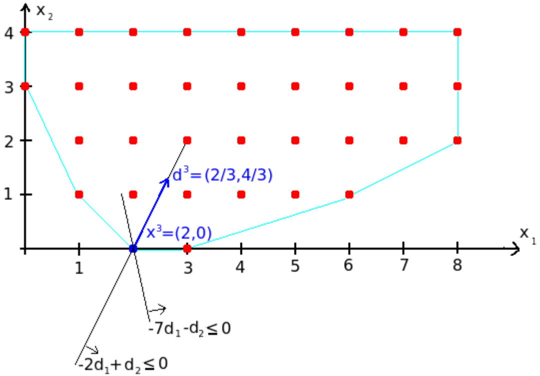
Figure: Iteration 2, feasible d cone ($(3,0)$ is apex), d^2 and x^2



Using x^2, we generate cut $d^\top(x^0 - x^2) \leq 0$, i.e. $-2d_1 + d_2 \leq 0$.

A Small Example: Iteration 3

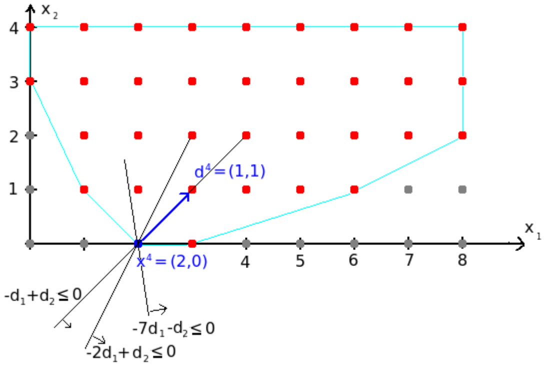
Figure: Iteration 3, feasible d cone, d^3 and x^3



Using x^3, we generate cut $d^\top(x^0 - x^3) \leq 0$, i.e. $-d_1 + d_2 \leq 0$.

A Small Example: Iteration 4

Figure: Iteration 4, feasible d cone, d^4 and x^4



Using x^4, we generate cut $d^\top(x^0 - x^4) \leq 0$, i.e. $-d_1 + d_2 \leq 0$. Note that current d^4 does not violate this cut, then d^4 is the optimal solution of inverse problem.