Bilevel Integer Programming

Ted Ralphs[1]

Joint work with:
Scott DeNegre[1], Menal Guzelsoy[2],
Andrea Lodi[3], Fabrizio Rossi[4], Stefano Smriglio[4]

[1]COR@L Lab, Department of Industrial and Systems Engineering, Lehigh University
[2]ISyE, Georgia Institute of Technology
[3]DEIS, Universitá di Bologna
[4]Dipartimento di Informatica, Universitá di L'Aquila

COR@L
COMPUTATIONAL OPTIMIZATION RESEARCH AT LEHIGH

Outline

Motivation

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

Nash and Stackelberg Games

- Many game theoretic models can be formulated as optimization problems involving multiple decision makers.
- In a *Nash game*, the players are treated as equals and take simultaneous action.
- Computationally, one often wishes to find a *Nash equilibrium*, in which the action of each player is optimal, given the actions of all other players.
- In a *Stackelberg game*, there is a dominant player, called the *leader*, who acts first and other players react.
- In this case, one is concerned with determining the leader's decision, given the assumption that the *followers* will react optimally.
- This can often be modeled as a bilevel program.

Some Caveats

- This talk covers a lot of territory, so a lot of details are left out.
- The talk contains a plethora of notation and uses almost every Greek letter in the alphabet.
- The talk is chock full of “half-baked” ideas, some very fresh.
- Because of the nature of the material, it is extremely difficult to stay consistent with max and min, the sense of various inequalities, etc.
- I will switch back and forth beween a standard form mixed integer linear program and a standard form bilevel mixed integer linear program.
- Please stop me and ask questions as we go along.

Bilevel (Integer) Linear Programming

Formally, a *bilevel linear program* is described as follows.

- $x \in X \subseteq \mathbb{R}^{n_1}$ are the *upper-level variables*
- $y \in Y \subseteq \mathbb{R}^{n_2}$ are the *lower-level variables*

Bilevel (Integer) Linear Program

$$\max \left\{ c^1 x + d^1 y \mid x \in \mathcal{P}_U \cap X, y \in \operatorname{argmin}\{ d^2 y \mid y \in \mathcal{P}_L(x) \cap Y \} \right\} \quad \text{(MIBLP)}$$

The *upper-* and *lower-level feasible regions* are:

$$\mathcal{P}_U = \left\{ x \in \mathbb{R}_+ \mid A^1 x \leq b^1 \right\} \text{ and}$$
$$\mathcal{P}_L(x) = \left\{ y \in \mathbb{R}_+ \mid G^2 y \geq b^2 - A^2 x \right\}.$$

We consider the general case in which $X = \mathbb{Z}^{p_1} \times \mathbb{R}^{n_1 - p_1}$ and $Y = \mathbb{Z}^{p_2} \times \mathbb{R}^{n_2 - p_2}$.

Notation

Notation

$$
\begin{aligned}
\Omega^I &= \{(x,y) \in (X \times Y) \mid x \in \mathcal{P}_U, y \in \mathcal{P}_L(x)\} \\
\Omega &= \{(x,y) \in (\mathbb{R}^{n_1} \times \mathbb{R}^{n_2}) \mid x \in \mathcal{P}_U, y \in \mathcal{P}_L(x)\} \\
M^I(x) &= \operatorname{argmin}\{d^2 y \mid y \in (\mathcal{P}_L(x) \cap Y)\} \\
\mathcal{F}^I &= \left\{(x,y) \mid x \in (\mathcal{P}_U \cap X), y \in M^I(x)\right\} \\
\mathcal{F} &= \left\{(x,y) \mid x \in \mathcal{P}_U, y \in \operatorname{argmin}\{d^2 y \mid y \in \mathcal{P}_L(x)\}\right\}
\end{aligned}
$$

- Underlying bilevel linear program (BLP):

$$\min_{(x,y) \in \mathcal{F}} c^1 x + d^1 y$$

- Underlying mixed integer linear program (MILP):

$$\min_{(x,y) \in \Omega^I} c^1 x + d^1 y$$

- Underlying linear program (LP):

$$\min_{(x,y) \in \Omega} c^1 x + d^1 y$$

(Standard) Mixed Integer Linear Programs

- In parts of the talk, we will need to consider a (standard) *mixed integer linear program* (MILP).
- To simplify matters, when we discuss a standard MILP, it will be of the form

MILP

$$\min\{c^\top x \mid x \in \mathcal{P} \cap (\mathbb{Z}^p \times \mathbb{R}^{n-p})\}, \qquad \text{(MILP)}$$

where $\mathcal{P} = \{x \in \mathbb{R}^n_+ \mid Ax = b\}$, $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$.

Basic Assumptions

- For the remainder of the talk, we consider decision problems in which there are two DMs, a *leader* or *upper-level* DM and a *follower* or *lower-level* DM.
- We assume *individual rationality* of the two DMs, i.e., the leader can predict the follower's reaction to a given course of action.
- For simplicity, we also assume that for every action by the leader, the follower has a feasible reaction.
  - The follower may in fact have more than one equally favorable reaction to a given action by the leader.
  - These alternatives may not be equally favorable to the leader.
  - We assume that the leader may choose among the follower's alternatives.
  - This assumption is reasonable if the players have a "semi-cooperative" relationship.
- We assume the feasible set $\mathcal{F}^I$ is nonempty and compact to ensure solutions exist.

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": leader represents external phenomena that cannot be controlled.
  - Weather
  - External market conditions
- Controlling optimized systems: follower represents a system that is optimized by its nature.
  - Electrical networks
  - Biological systems

Bilevel Structure in Branch and Cut

- Consider again the instance (MILP).
- A *branch-and-cut algorithm* to solve this problem requires the solution of two fundamental decision problems.

Definition 1 *The separation problem for a polyhedron $\mathcal{Q}$ is to determine for a given $\hat{x} \in \mathbb{R}^n$ whether or not $\hat{x} \in \mathcal{Q}$ and if not, to produce an inequality $(\bar{\alpha}, \bar{\beta}) \in \mathbb{R}^{n+1}$ valid for $\mathcal{Q}$ and for which $\bar{\alpha}^\top \hat{x} < \bar{\beta}$.*

Definition 2 *The branching problem for a set $\mathcal{S}$ is to determine for a given $\hat{x} \in \mathbb{R}^n$ whether $\hat{x} \in \mathcal{S}$ and if not, to produce a disjunction*

$$\bigvee_{h \in \mathcal{Q}} A^h x \geq b^h, \; x \in \mathcal{S} \tag{1}$$

that is satisfied by all points in $\mathcal{S}$, but not satisfied by $\hat{x}$.

Bilevel Structure of the Separation Problem

- Often, we wish to select an inequality that maximizes violation, i.e.,

$$(\bar{\alpha}, \bar{\beta}) \in \text{argmin}_{(\alpha,\beta)\in\mathbb{R}^{n+1}}\{\alpha^\top \hat{x} - \beta \mid \alpha^\top x \geq \beta \; \forall x \in \mathcal{Q}\} \tag{2}$$

- To make the problem tractable, we may restrict ourselves to a specific *template class* of valid inequalities with well-defined structure.
- Given a class $\mathcal{C}$, calculation of the right-hand side β required to ensure (α, β) is a member of $\mathcal{C}$ may itself be an optimization problem.
- The separation problem for the class $\mathcal{C}$ with respect to a given $\hat{x} \in \mathbb{R}^n$ can in principle be formulated as the bilevel program:

$$\begin{aligned} \min \;\; & \alpha^\top \hat{x} - \beta && (3)\\ & \alpha \in \mathcal{C}_\alpha && (4)\\ & \beta = \min\{\alpha^\top x \mid x \in \mathcal{F}\}, && (5) \end{aligned}$$

where the set $\mathcal{C}_\alpha \subseteq \mathbb{R}^n$ is the projection of $\mathcal{C}$ into the space of coefficient vectors and $\mathcal{F}$ is the closure over the class $\mathcal{C}$.

Example: Disjunctive cuts

- Given a MIP in the form (MILP), **?** showed how to derive a valid inequality by exploiting any fixed disjunction

$$\pi^\top x \leq \pi_0 \quad \text{OR} \quad \pi^\top x \geq \pi_0 + 1 \; \forall x \in \mathbb{R}^n, \tag{6}$$

  where $\pi \in \mathbb{Z}^n$ and $\pi_0 \in \mathbb{Z}$.
- A *disjunctive inequality* is one valid for the convex hull of union of $\mathcal{P}_1$ and $\mathcal{P}_2$, obtained by imposing the two terms of the disjunction.
- Conceptually, the separation problem can be written as the following bilevel program:

$$\begin{aligned}
\min \quad & \alpha^\top \hat{x} - \beta && (7)\\
& \alpha \geq u^\top A - u_o \pi && (8)\\
& \alpha \geq v^\top A + v_o \pi && (9)\\
& u, v, u_0, v_0 \geq 0 && (10)\\
& u_0 + v_0 = 1 && (11)\\
& \beta = \min\{\alpha^\top x \mid x \in \mathcal{P}_1 \cup \mathcal{P}_2\} && (12)
\end{aligned}$$

Example: Disjunctive Cuts (cont.d)

- Equation (12) requires β to have the largest value consistent with validity.
- To ensure the cut is valid, we need only ensure that

$$\beta \leq \min\{u^\top b - u_0\pi_0, v^\top b + v_0(\pi_0 + 1)\}. \tag{13}$$

- Using the standard modeling trick, we can rewrite (13) as

$$\beta \leq u^\top b - u_0\pi_0 \tag{14}$$
$$\beta \leq v^\top b + v_0(\pi_0 + 1). \tag{15}$$

- The sense of the optimization ensures that (13) holds at equality.

Example: Capacity Constraints for CVRP

- In the Capacitated Vehicle Routing Problem (CVRP), the *capacity constraints* are of the form

$$\sum_{\substack{e=\{i,j\}\in E \\ i\in S, j\notin S}} x_e \geq 2b(S) \quad \forall S \subset N,\ |S| > 1, \tag{16}$$

  where $b(S)$ is any lower bound on the number of vehicles required to serve customers in set S.

- By defining binary variables
  - $y_i = 1$ if customer i belongs to $\bar{S}$, and
  - $z_e = 1$ if edge e belongs to $\delta(\bar{S})$,

  we obtain the following bilevel formulation for the separation problem:

$$\min \sum_{e\in E} \hat{x}_e z_e - 2b(\bar{S}) \tag{17}$$

$$z_e \geq y_i - y_j \qquad \forall e \in E \tag{18}$$

$$z_e \geq y_j - y_i \qquad \forall e \in E \tag{19}$$

$$b(\bar{S}) = \max\{\bar{S} \mid b(\bar{S}) \text{ is a valid lower bound}\} \tag{20}$$

Example: Capacity Constraints for CVRP (cont.d)

If the bin packing problem is used in the lower-level, the formulation becomes:

$$\min \sum_{e \in E} \hat{x}_e z_e - 2b(\bar{S}) \tag{21}$$

$$z_e \geq y_i - y_j \qquad \forall e = \{i, j\} \tag{22}$$

$$z_e \geq y_j - y_i \qquad \forall e = \{i, j\} \tag{23}$$

$$b(\bar{S}) = \min \sum_{\ell=1}^{n} h_\ell \tag{24}$$

$$\sum_{\ell=1}^{n} w_i^\ell = y_i \qquad \forall i \in N \tag{25}$$

$$\sum_{i \in N} d_i w_i^\ell \leq K h_\ell \qquad \ell = 1, \ldots, n, \tag{26}$$

where we introduce the additional binary variables

- $w_i^\ell = 1$ if customer i is served by vehicle ℓ, and
- $h_\ell = 1$ if vehicle ℓ is used.

Bilevel Structure of the Branching Problem

- A typical criteria for selecting a branching disjunction is to maximize the bound increase resulting from imposing the disjunction.
- The problem of selecting the disjunction whose imposition results in the largest bound improvement has a natural *bilevel structure*.
  - The upper-level variables can be used to model the choice of disjunction (we'll see an example shortly).
  - The lower-level problem models the bound computation after the disjunction has been imposed.
- In strong branching, we are solving this problem essentially by enumeration.
- The bilevel branching paradigm is to select the branching disjunction directly by solving a bilevel program.

Example: Interdiction Branching

The following is a bilevel programming formulation for the problem of finding a smallest branching set in interdiction branching:

$$
\begin{aligned}
\text{(BBP)} \quad & \max \textstyle\sum c^\top x \\
\text{s.t.} \quad & \\
& c^\top x \leq \bar{z} \\
& \\
& y \in \mathbb{B}^n \\
& \\
& x \in \arg\max_x c^\top x \\
& \text{s.t.} \\
& \qquad x_i + y_i \leq 1, \quad i \in N^a \\
& \\
& \qquad x \in \mathcal{F}^a
\end{aligned}
$$

where $\mathcal{F}$ is the feasible region of a given relaxation of the original problem used for computing the bound.

Recourse Problems

- If $d^1 = -d^2$, we can view this as a *mathematical program with recourse*.
- We can reformulate the bilevel program as follows.

$$\min\{-c^1 x + Q(x) \mid x \in \mathcal{P}_U \cap X\}, \tag{27}$$

where

$$Q(x) = \min\{d^1 y \mid y \in \mathcal{P}_L(x) \cap Y\}. \tag{28}$$

- The function Q is known as the *value function* of the recourse problem.

Recourse Problems with Continuous Second Stage

If $Y = \mathbb{R}^{n_1}$, it is well-known by LP duality that

$$Q(x) = \max\{u(b^2 - A^2x) \mid uG^2 \leq d^1, u \in \mathbb{R}^{m_2}_+\}, \tag{29}$$

so we can further reformulate (27) as

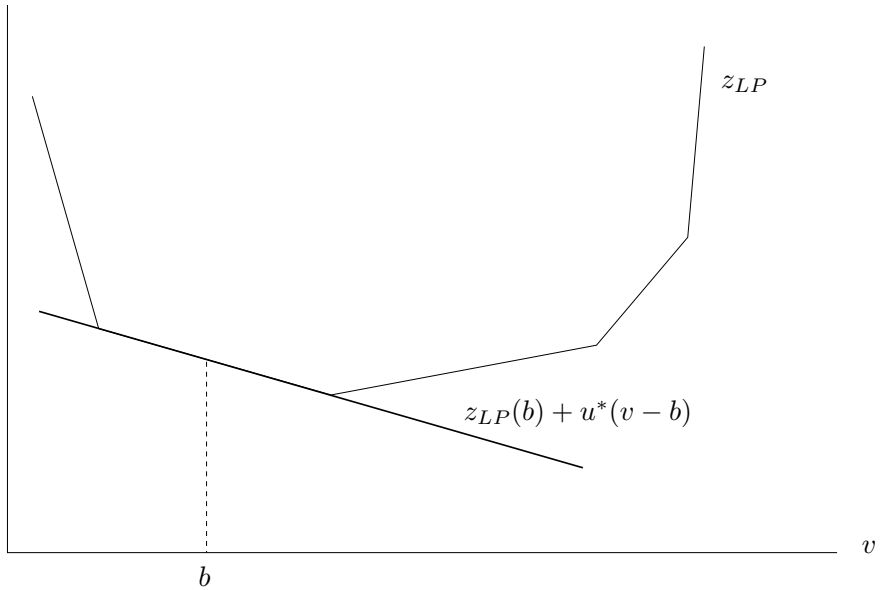
$$\min\{c^1x + z \mid x \in \mathcal{P}_U \cap X, z \geq u^i(b^2 - A^2x)\ \forall i \in \mathcal{D}\}, \tag{30}$$

where $\mathcal{D}$ is a set indexing the extreme points of the dual polyhedron

$$\{u \in \mathbb{R}^{m_2}_+ \mid uG^2 \geq d^1\}. \tag{31}$$

Benders Decomposition

- We can then solve (27) by Benders decomposition.
- This amounts to solving (30) by cut generation.
- The value function of the lower-level problem is convex in the upper-level variables, which is what makes the problem tractable.



Two-Stage Stochastic Integer Programs

- Consider the two-stage stochastic mixed integer program

$$\min\{c^1 x + \mathbb{E}_\xi Q_\xi(x) \mid x \in \mathcal{P}_U \cap X\}, \tag{32}$$

where

$$Q_\xi(x) = \min\{d^2 y \mid y \in Y, G^2 y \geq \omega(\xi) - A^2 x\}, \tag{33}$$

ξ is a random variable from a probability space $(\Xi, \mathscr{F}, \mathcal{P})$, and for each $\xi \in \Xi$, $\omega(\xi) \in \mathbb{R}^{m_2}$.

- If the distribution of ξ is discrete and has finite support, then (32) is a bilevel program.

Continuous Second Stage

- In general, if $Y = \mathbb{R}^{n_1}$, then the lower-level problem can be replaced with its optimality conditions.
- The optimality conditions for the lower-level optimization problem are

$$\begin{aligned} G^2 y &\geq b^2 - A^2 x \\ uG^2 &\leq d^2 \\ u(b^2 - G^2 - A^2 x) &= 0 \\ (d^2 - uG^2)y &= 0 \\ u, y &\in \mathbb{R}_+ \end{aligned}$$

- When $X = \mathbb{R}^{n_1}$, this is a special case of a class of non-linear mathematical programs known as *mathematical programs with equilibrium constraints* (MPECs).
- MPECs can be solved in a number of ways, including converting it to a standard integer program.
- Note that in this case, the value function of the lower-level problem is piecewise linear, but not necessarily convex.

Other Cases

- Pure integer.
- Positive constraint matrix at lower level.
- Binary variables at the upper and/or lower level.
- *Interdiction problems*.

Mixed Integer Interdiction

$$\max_{x \in \mathcal{P}_U^I} \min_{y \in \mathcal{P}_L^I(x)} dy \qquad \text{(MIPINT)}$$

where

$$\mathcal{P}_U^I = \left\{ x \in \mathbb{B}^n \mid A^1 x \leq b^1 \right\}$$
$$\mathcal{P}_L^I(x) = \left\{ y \in \mathbb{Z}^p \times \mathbb{R}^{n-p} \mid G^2 y \geq b^2, y \leq u(e - x) \right\}.$$

- 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 lower-level systems described by general MILPs.

The General Case

- When some/all of the variables are discrete, the problem becomes more complex, both practically and theoretically.
- Many of the ideas from these special cases can be extended *in principle*.
- However, it's very difficult to obtain anything tractable *in practice*.
- An obvious question is whether we can generalize any of the very well-developed methodology we know for solving standard MILPs.
- We can do so to a certain extent, but the proper generalizations are not obvious.

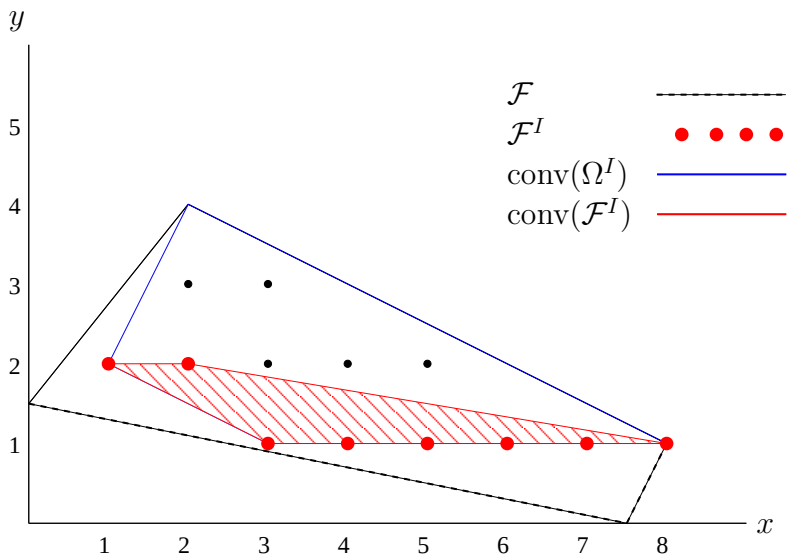
Complexity

- It is perhaps to be expected that general BMILPs are in a different complexity class than standard MILPs.
- This is because checking feasibility is itself an NP-complete problem for BMILPs.
- Even the case in which $X = \mathbb{R}^n_1$ and $Y = \mathbb{R}^n_2$ is NP-complete.
- We conjecture that bilevel programming is Σ^p_2-complete.
- The class Σ^p_2 is one step higher in the so-called "polynomial-time hierarchy" than the class NP($= \Sigma^p_1$).
- Roughly speaking, this class consists of problems that could be solved in nondeterministic polynomial time, given an oracle for problems in NP.

Example

Consider the following instance of (MIBLP) from **?**.

$$\begin{aligned} \min_{x \in \mathbb{Z}} \quad & -x - 10y \\ \text{subject to} \quad & y \in \operatorname{argmin} \{ y : 25x - 20y \geq -30 \\ & \qquad -x - 2y \geq -10 \\ & \qquad -2x + y \geq -15 \\ & \qquad 2x + 10y \geq 15 \\ & \qquad y \in \mathbb{Z} \ \} \end{aligned}$$



From the figure, we can make several observations:

1. $\mathcal{F} \subseteq \Omega$, $\mathcal{F}^I \subseteq \Omega^I$, and $\Omega^I \in \Omega$
2. $F^I \not\subseteq \mathcal{F}$
3. Solutions to (MIBLP) do not occur at extreme points of $\text{conv}(\Omega^I)$

Properties of MIBLPs

In this example:

- Optimizing over $\mathcal{F}$ yields the *integer* solution $(8, 1)$, with the upper-level objective value 18.
- Imposing integrality yields the solution $(2, 2)$, with upper-level objective value 22

From this we can make two important observations:

1. The objective value obtained by relaxing integrality is not a valid bound on the solution value of the original problem,
2. Even when solutions to $\max_{(x,y)\in\mathcal{F}} c^1x + d^1y$ are in $\mathcal{F}^I$, they are not necessarily optimal.

Thus, some familiar properties from the MILP case do not hold here.

Understanding the Structure

- The key to understanding the structure of an MIBLP is to consider the value function associated with the lower-level problem.
- This *value function* of the instance (MILP) is a function $z : \mathbb{R} \to \mathbb{R} \cup \{\pm\infty\}$ defined as follows:

MILP Value Function

$$z(d) = \min_{x \in S(d)} c^\top x, \tag{34}$$

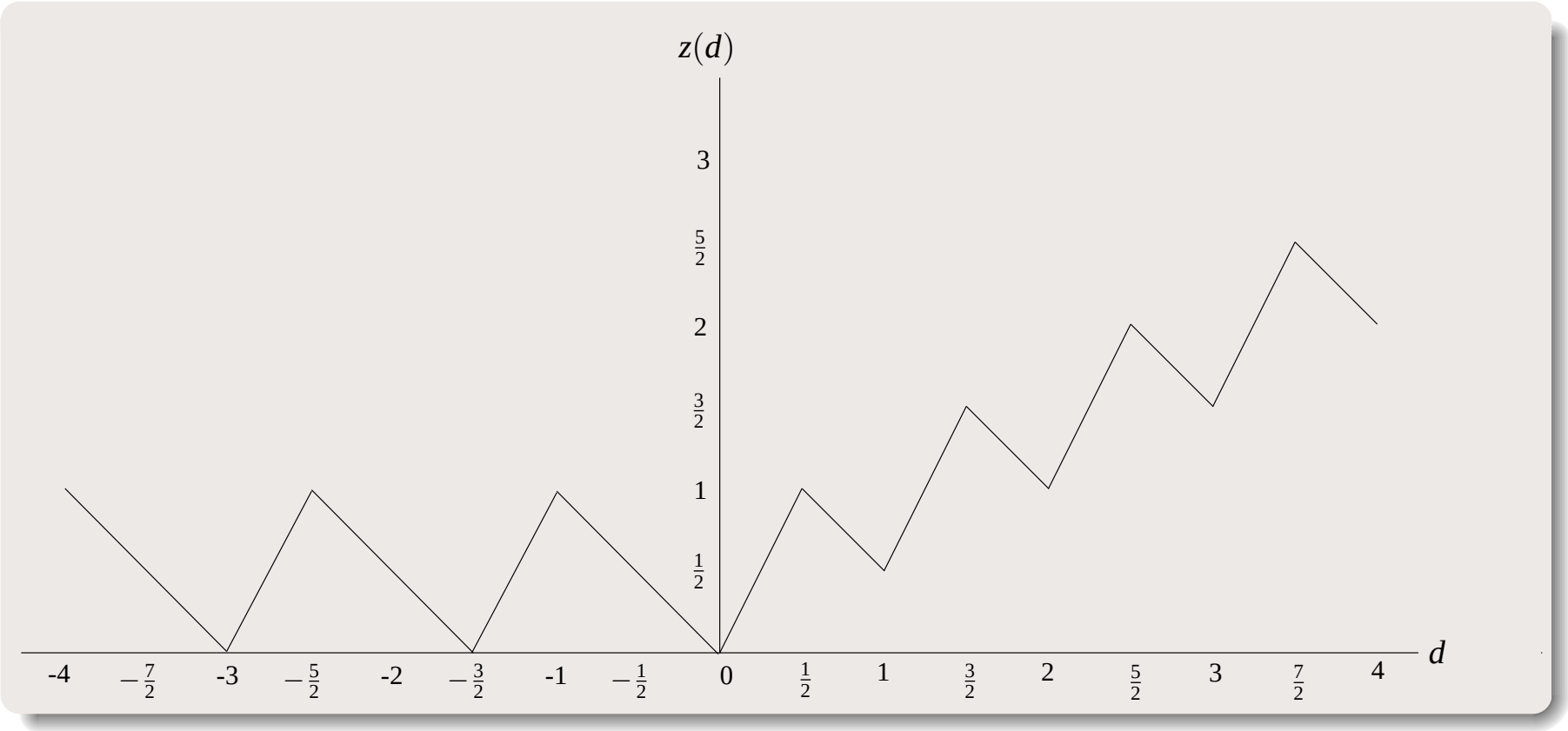
where, for a given right-hand side vector $d \in \mathbb{R}^m$,

$$S(d) = \{x \in \mathbb{Z}^p_+ \times \mathbb{R}^{n-p}_+ \mid Ax = d\}.$$

- In what follows, we will set $\Theta = \{d \in \mathbb{R}^m \mid S(d) \neq \emptyset\}$.

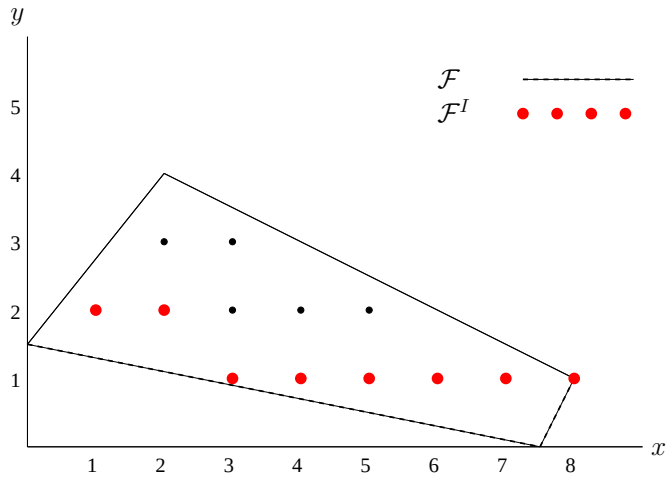
Example: Value Function

$$\begin{aligned} z_{IP} = \min \quad & \tfrac{1}{2}x_1 + 2x_3 + x_4 \\ \text{s.t} \quad & x_1 - \tfrac{3}{2}x_2 + x_3 - x_4 = b \quad \text{and} \\ & x_1, x_2 \in \mathbb{Z}_+, x_3, x_4 \in \mathbb{R}_+. \end{aligned}$$



Value Function Reformulation

If we knew the value function explicitly, we could reformulate (MIBLP) as



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

where z_{LL} is the value function of the lower-level problem.

- This is, in principle, a standard mathematical program.
- It is easy to see why relaxing integrality does not yield a valid bound.
- In this case, we are effectively replacing z_{LL} with the value function of the LP relaxation (more on this soon).

Properties of the Value Function

- It is subadditive over Θ.
- It is piecewise polyhedral.
- For an ILP, it can be computed by a finite number of limited operations on elements of the RHS:

(i) rational multiplication
(ii) nonnegative combination
(iii) rounding
(iv) taking the maximum

} *Chvátal fcns.* } *Gomory fcns.*

Properties of the Value Function (cont.)

- There is a one-to-one correspondence between ILP instances and Gomory functions.
- The *Jeroslow Formula* shows that the value function of an MILP can be constructed from the value function of an associated ILP.
- The value function of the earlier example is

$$z(d) = \min \left\{ \begin{array}{l} \frac{3}{2} \max \left\{ \left\lceil \frac{\lfloor 2d \rfloor}{3} \right\rceil, \left\lceil \frac{\lfloor 2d \rfloor}{2} \right\rceil \right\} + \frac{3\lceil 2d \rceil}{2} + 2d, \\ \frac{3}{2} \max \left\{ \left\lceil \frac{\lceil 2d \rceil}{3} \right\rceil, \left\lceil \frac{\lceil 2d \rceil}{2} \right\rceil \right\} - d, \end{array} \right\}$$

Jeroslow Formula

We consider again the instance (MILP). Let the set $\mathscr{E}$ consist of the index sets of dual feasible bases of the linear program

$$\min\{\frac{1}{M}c_C x_C \ : \ \frac{1}{M}A_C x_C = b, x \geq 0\}$$

where $M \in \mathbb{Z}_+$ such that for any $E \in \mathscr{E}$, $MA_E^{-1}a^j \in \mathbb{Z}^m$ for all $j \in I$.

Jeroslow Formula

Theorem 1 *There is a $g \in \mathscr{G}^m$ such that*

$$z(d) = \min_{E \in \mathscr{E}} g(\lfloor d \rfloor_E) + v_E(d - \lfloor d \rfloor_E) \ \ \forall d \in \mathbb{R}^m \ \ \textit{with} \ \ \mathcal{S}(d) \neq \emptyset,$$

where for $E \in \mathscr{E}$, $\lfloor d \rfloor_E = A_E \lfloor A_E^{-1} d \rfloor$ and v_E is the corresponding basic feasible solution.

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 upper or lower bounding functions

Lower bounds

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

Upper bounds

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

The LP Relaxation

- The value function of the LP relaxation of the original problem provides an easy lower bound.

$$F_{LP}(d) = \max_{v \in \mathbb{R}^m} \{vd : vA \leq c\}.$$

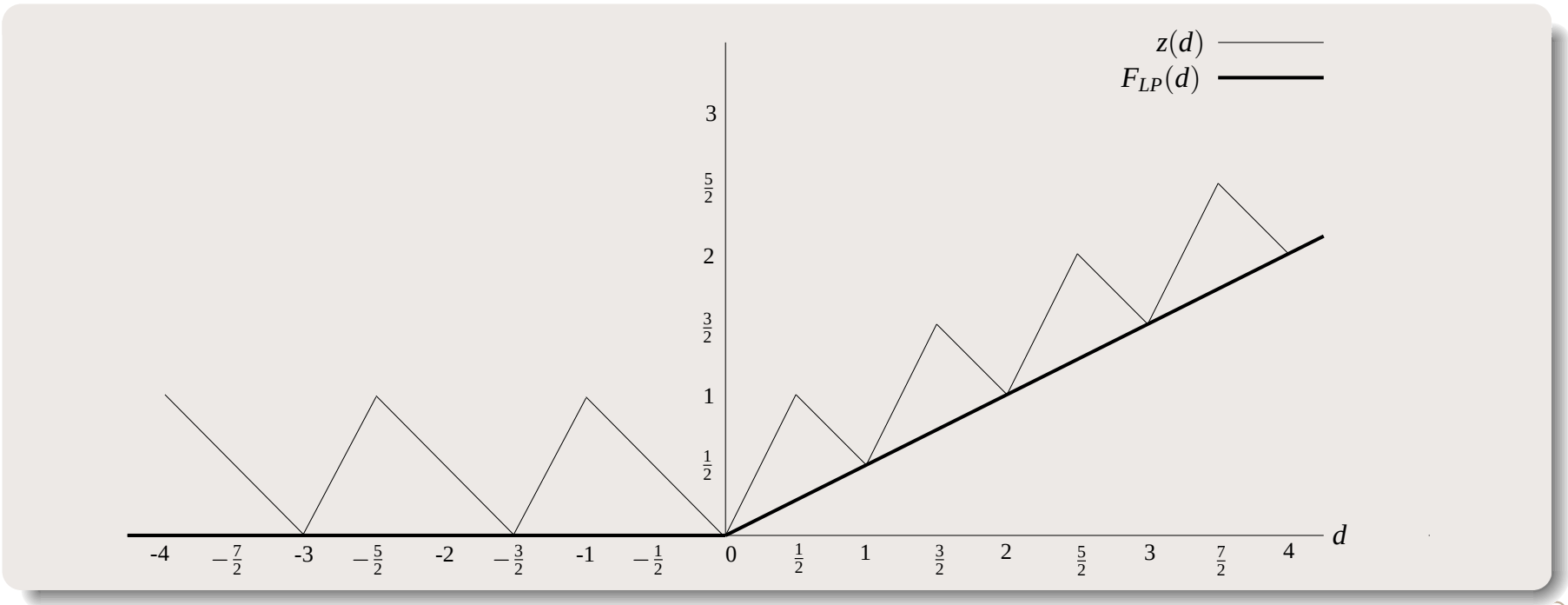
- By linear programming duality theory, we have $F_{LP}(d) \leq z(d)$ for all $d \in \mathbb{R}^m$.
- Of course, F_{LP} is not necessarily strong.

Example: LP Dual Function

$$\begin{aligned} F_{LP}(d) = \min \quad & vd, \\ \text{s.t} \quad & 0 \geq v \geq -\tfrac{1}{2}, \text{ and} \\ & v \in \mathbb{R}, \end{aligned}$$

which can be written explicitly as

$$F_{LP}(d) = \begin{cases} 0, & d \leq 0 \\ -\frac{1}{2}d, & d > 0 \end{cases}.$$



Relaxing Linear Constraints

Consider the value functions of each single row relaxation:

$$z_i(q) = \min\{cx \mid a_i x = q, x \in \mathbb{Z}^r_+ \times \mathbb{R}^{n-r}_+\}\ q \in \mathbb{R}, i \in M \equiv \{1, \ldots, m\}$$

where a_i is the i^{th} row of A.

Theorem 2 *Let* $F(d) = \max_{i \in M}\{z_i(d_i)\},\ d = (d_1, \ldots, d_m),\ d \in \mathbb{R}^m$. *Then* F *is subadditive and* $F(d) \leq z(d)\ \forall d \in \mathbb{R}^m$.

We know a lot about the structure of the value function of single-row relaxations.

Aggregation

For $S \subseteq M$, $\omega \in \mathbb{R}^{|S|}$, set

$$G_S(q,\omega) = \min\{cx \mid \omega a_S x = \omega q, x \in \mathbb{Z}^r_+ \times \mathbb{R}^{n-r}_+\} \; \forall q \in \mathbb{R}^{|S|}$$

Theorem 3 *Let*

$$F_S(\omega,d) = \max\left\{G_S(d_S,\omega), \max_{i\in M\setminus S}\{z_i(d_i)\}\right\}, \; d \in \mathbb{R}^m.$$

F_S is subadditive and $z(d) \geq F_S(\omega,d)$ for any $\omega \in \mathbb{R}^{|S|}$, $d \in \mathbb{R}^m$.

As with cutting planes, different aggregation procedures are possible.

Dual Functions

- A *dual function* $F : \mathbb{R}^m \rightarrow \mathbb{R}$ is one that satisfies $F(d) \leq z(d)$ for all $d \in \mathbb{R}^m$.
- This is a generalization of the concept of dual solution from the LP case.
- How to select such a function?
- We generally choose one for which $F(b) \approx z(b)$ for some particular $b \in \mathbb{R}^{m_2}$ of interest.
- This results in the following dual problem:

$$z_D = \max \{F(b) : F(d) \leq z(d),\ d \in \mathbb{R}^m, F \in \Upsilon^m\}$$

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

- We call F^* *strong* for this instance if F^* is a *feasible* dual function and $F^*(b) = z(b)$.
- This dual instance always has a solution F^* that is strong if the value function is bounded and $\Upsilon^m \equiv \{f \mid f : \mathbb{R}^m \rightarrow \mathbb{R}\}$. Why?

The Subadditive Dual

By considering that

$$\begin{aligned} F(d) \leq z(d),\ d \in \mathbb{R}^m \quad &\Longleftrightarrow \quad F(d) \leq cx\ ,\ x \in \mathcal{S}(d),\ d \in \mathbb{R}^m \\ &\Longleftrightarrow \quad F(Ax) \leq cx\ ,\ x \in \mathbb{Z}^n_+, \end{aligned}$$

the generalized dual problem can be rewritten as

$$z_D = \max\{F(b) : F(Ax) \leq cx,\ x \in \mathbb{Z}^r_+ \times \mathbb{R}^{n-r}_+,\ F \in \Upsilon^m\}.$$

Can we further restrict Υ^m and still guarantee a strong dual solution?

- The class of linear functions? NO!
- The class of convex functions? NO!
- The class of sudadditive functions? YES!

The Subadditive Dual

- Let a function F be defined over a domain V. Then F is subadditive if $F(v_1) + F(v_2) \geq F(v_1 + v_2) \forall v_1, v_2, v_1 + v_2 \in V$.
- Note that the value function z is subadditive over Θ. Why?
- If $\Upsilon^m \equiv \Gamma^m \equiv \{F \text{ is subadditive} \mid F : \mathbb{R}^m \rightarrow \mathbb{R}, F(0) = 0\}$, we can rewrite the dual problem above as the *subadditive dual*

$$\begin{aligned} z_D = \max \quad & F(b) \\ & F(a^j) \leq c_j \quad j = 1, ..., r, \\ & \bar{F}(a^j) \leq c_j \quad j = r+1, ..., n, \text{ and} \\ & F \in \Gamma^m, \end{aligned}$$

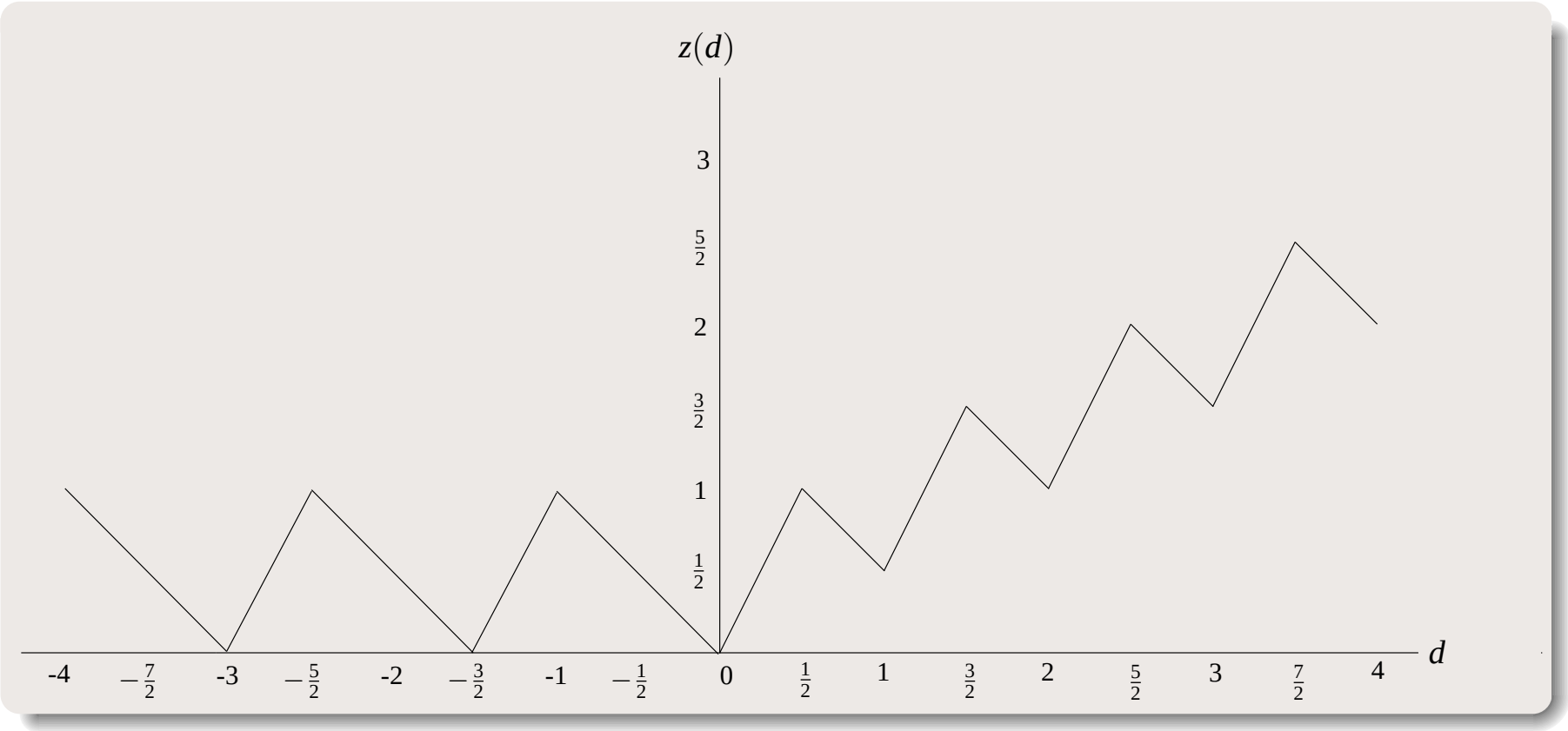
where the function $\bar{F}$ is defined by

$$\bar{F}(d) = \limsup_{\delta \rightarrow 0^+} \frac{F(\delta d)}{\delta} \quad \forall d \in \mathbb{R}^m.$$

- Here, $\bar{F}$ is the *upper d-directional derivative* of F at zero.

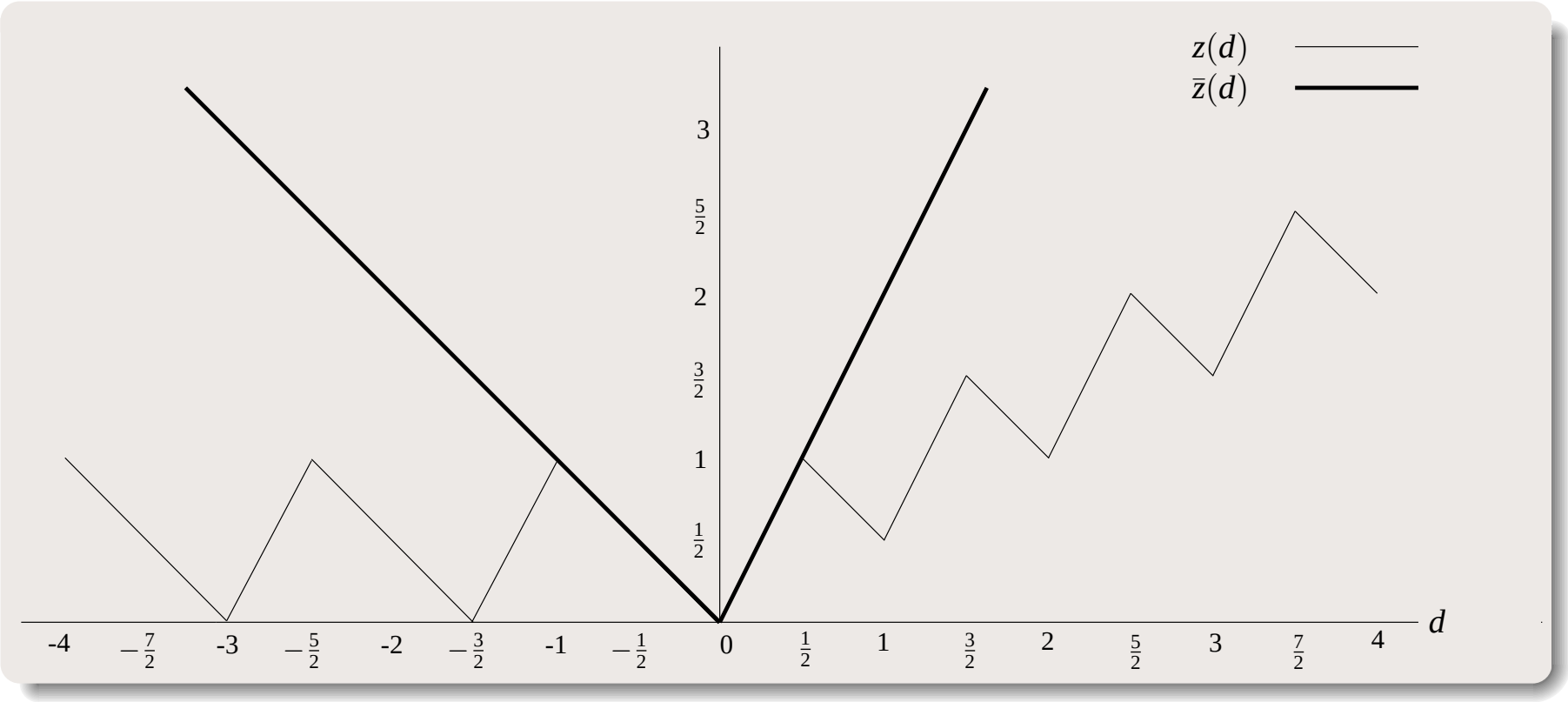
Example: Value Function Revisited

$$
\begin{aligned}
z_{IP} = \min \quad & \tfrac{1}{2}x_1 + 2x_3 + x_4 \\
\text{s.t} \quad & x_1 - \tfrac{3}{2}x_2 + x_3 - x_4 = b \quad \text{and} \\
& x_1, x_2 \in \mathbb{Z}_+, x_3, x_4 \in \mathbb{R}_+.
\end{aligned}
$$



Example: Upper D-directional Derivative

- The upper d-directional derivative can be interpreted as the slope of the value function in direction d at 0.
- For the example, we have



Example: Subadditive Dual

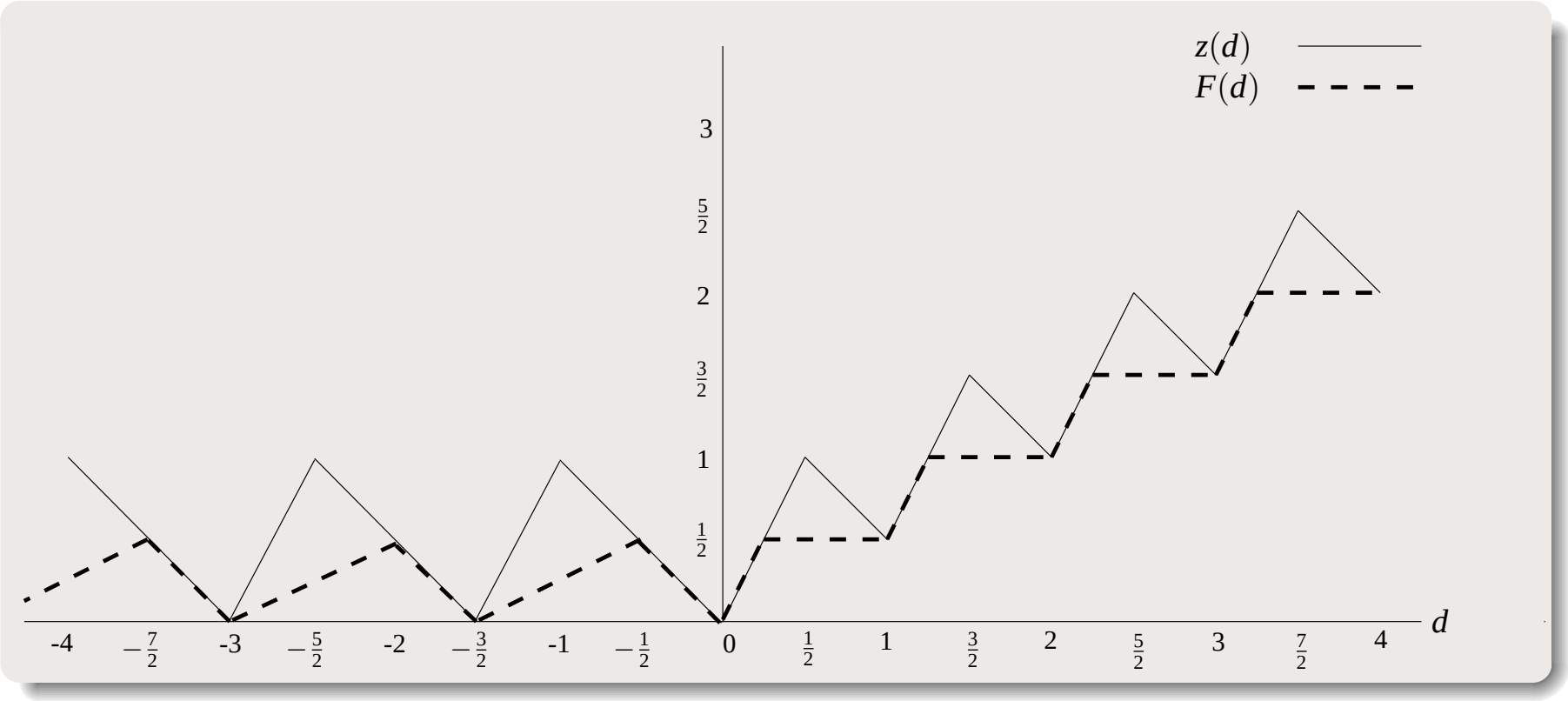
For our IP instance, the subadditive dual problem is

$$\begin{array}{lrl} \max & F(b) & \\ & F(1) & \leq \frac{1}{2} \\ & F(-\frac{3}{2}) & \leq 0 \\ & \bar{F}(1) & \leq 2 \\ & \bar{F}(-1) & \leq 1 \\ & F \in \Gamma^1. & \end{array} .$$

and we have the following feasible dual functions:

1. $F_1(d) = \frac{d}{2}$ is an optimal dual function for $b \in \{0, 1, 2, ...\}$.
2. $F_2(d) = 0$ is an optimal function for $b \in \{..., -3, -\frac{3}{2}, 0\}$.
3. $F_3(d) = \max\{\frac{1}{2}\lceil d - \frac{\lceil [d] - d \rceil}{4} \rceil, 2d - \frac{3}{2}\lceil d - \frac{\lceil [d] - d \rceil}{4} \rceil\}$ is an optimal function for $b \in \{[0, \frac{1}{4}] \cup [1, \frac{5}{4}] \cup [2, \frac{9}{4}] \cup ...\}$.
4. $F_4(d) = \max\{\frac{3}{2}\lceil \frac{2d}{3} - \frac{2\lceil \lceil \frac{2d}{3} \rceil - \frac{2d}{3} \rceil}{3} \rceil - d, -\frac{3}{4}\lceil \frac{2d}{3} - \frac{2\lceil \lceil \frac{2d}{3} \rceil - \frac{2d}{3} \rceil}{3} \rceil + \frac{d}{2}\}$ is an optimal function for $b \in \{... \cup [-\frac{7}{2}, -3] \cup [-2, -\frac{3}{2}] \cup [-\frac{1}{2}, 0]\}$

Example: Feasible Dual Functions



- Notice how different dual solutions are optimal for some right-hand sides and not for others.
- Only the value function is optimal for all right-hand sides.

Properties of the Subadditive Dual

Using the subadditive dual, we can generalize many of the properties of the LP dual.

- Weak/Strong Duality
- Farkas Lemma
- Optimality conditions (complementary slackness)

Reformulation with Optimality Conditions

In principle, we can use subadditive duality to obtain optimality conditions for the lower-level problem (reformulation shown here is for the pure integer case).

$$
\begin{aligned}
\max_{x,y,F} \quad & c^1 x + d^1 y \\
\text{subject to} \quad & A^1 x \leq b^1 \\
& A^2 x + G^2 y \geq b^2 \\
& F(g_j^2) \leq d_j^2, \quad \forall j = 1, \ldots, n_2 \\
& (F(g_j^2) - d_j^2) y_j = 0, \quad \forall j = 1, \ldots, n_2 \\
& \sum_{j=1}^{n_2} F(g_j^2) y_j = F(b^2 - A^2 x) \\
& x \in \mathbb{Z}_+^{n_1}, y \in \mathbb{Z}_+^{n_2}, F \in \Gamma^{m_2}.
\end{aligned}
$$

This is analogous to the reformulation in the continuous case, but is intractable in general.

Constructing Dual Functions

- Explicit construction
  - The Value Function
  - Generating Functions
- Relaxations
  - Lagrangian Relaxation
  - Quadratic Lagrangian Relaxation
  - Corrected Linear Dual Functions
- Primal Solution Algorithms
  - Cutting Plane Method
  - Branch-and-Bound Method
  - Branch-and-Cut Method

For the remainder of this part of the talk, we consider the instance (MILP).

Gomory's Procedure

- There is a Chvátal function that is optimal to the subadditive dual of an ILP with RHS $b \in \Omega_{IP}$ and $z_{IP}(b) > -\infty$.
- The procedure:
  In iteration k, we solve the following LP

$$\begin{aligned} z_{IP}(b)^{k-1} = \min \quad & cx \\ \text{s.t.} \quad & Ax = b \\ & \textstyle\sum_{j=1}^{n} f^i(a_j)x_j \geq f^i(b) \qquad i = 1, \ldots, k-1 \\ & x \geq 0 \end{aligned}$$

- The k^{th} cut, $k > 1$, is dependent on the RHS and written as:

$$f^k(d) = \left\lceil \sum_{i=1}^{m} \lambda_i^{k-1} d_i + \sum_{i=1}^{k-1} \lambda_{m+i}^{k-1} f^i(d) \right\rceil \quad \text{where} \quad \lambda^{k-1} = (\lambda_1^{k-1}, \ldots, \lambda_{m+k-1}^{k-1}) \geq 0$$

Gomory's Procedure (cont.)

- Assume that $b \in \Omega_{IP}$, $z_{IP}(b) > -\infty$ and the algorithm terminates after $k+1$ iterations.
- If u^k is the optimal dual solution to the LP in the final iteration, then

$$F^k(d) = \sum_{i=1}^{m} u_i^k d_i + \sum_{i=1}^{k} u_{m+i}^k f^i(d),$$

is a Chvátal function with $F^k(b) = z_{IP}(b)$ and furthermore, it is optimal to the subadditive ILP dual problem.

Gomory's Procedure (cont.)

Example: Let $b = 3$. At first iteration, we add the constraint

$$\lceil 2/2 \rceil x_1 + \lceil -2/2 \rceil x_2 + \lceil 1/2 \rceil x_3 + \lceil -1/2 \rceil x_4 \geq \lceil 3/2 \rceil$$

from the weight $\lambda_1 = 1/2$, i.e., the cut $x_1 - x_2 + x_3 \geq 2$. After resolving, we get an integer primal solution with the dual solution $u = (0, 1)$. Then the corresponding optimal dual function is:

$$F^1(d) = 0d + 1\lceil d/2 \rceil = \lceil d/2 \rceil$$

What does this mean?

Branch-and-Bound Method

- Assume that the primal problem is solved to optimality.
- Let T be the set of leaf nodes of the search tree.
- Thus, we've solved the LP relaxation of the following problem at node $t \in T$

$$z^t(b) = \begin{array}{ll} \min & cx \\ \text{s.t} & x \in \mathcal{S}_t(b) \end{array},$$

  where $\mathcal{S}_t(b) = \{Ax = b, x \geq l^t, -x \geq -u^t, x \in \mathbb{Z}^n\}$ and $u^t,\ l^t \in \mathbb{Z}^r$ are the branching bounds applied to the integer variables.
- Let $(v^t, \underline{v}^t, \bar{v}^t)$ be
  - the dual feasible solution used to prune node t, if t is feasibly pruned
  - a dual feasible solution (that can be obtained from it parent) to node t, if t is infeasibly pruned

Then,

$$F_{BB}(d) = \min_{t \in T} \{v^t d + \underline{v}^t l^t - \bar{v}^t u^t\}$$

is an optimal solution to the generalized dual problem.

Aggregated Lower Bounding Approximations

- Note that we can combine dual functions to better approximate the value function.
- This is similar to what is done to approximate the LP value function in Bender's Decomposition.
- $\mathcal{F}_1$: the set of dual functions obtained from single row relaxations
- $\mathcal{F}_2$: the set of the dual functions obtained from primal solution procedures for each $b \in U \subset \mathbb{R}^m$ where U is some collection of right-hand sides.
- Then, we call the dual function F defined as

$$F(d) = \max_{f \in \mathcal{F}_1 \cup \mathcal{F}_2} \{f(d)\} \quad \forall d \in \mathbb{R}^m$$

a global approximation of the value function.

Upper Bounding Approximations

- Just like the lower bounding approximations, we are interested in a function that is a valid upper bound for the value function.
- Since upper bounds are closely related to feasibility, it is harder to obtain such a function than to obtain dual/lower bounding functions.
- One possible way is to consider the maximal subadditive extension result.
- If $f(d) \geq z(d)$ forall $d \in [0, q]$, then any extension of f from $[0, q]$ to $\mathbb{R}^m_+$ is an upper bounding function.

Upper Bounding Approximations (cont.)

- Another way is to consider the continuous relaxation of the primal instance.
  - Assume that $\{u \in \mathbb{R}^m \mid uA_C \leq c_C\}$ is not empty and bounded.
  - Let $z_C(d) = \max\{vd \mid v \in V\}$ where V is the set of extreme points of dual polytope.
  - Then, $z_C(d) \geq z(d) \ \ \forall d \in \mathbb{R}^m$.
- Furthermore, we can move z_C to a given right hand side b to obtain strong upper bounding functions.

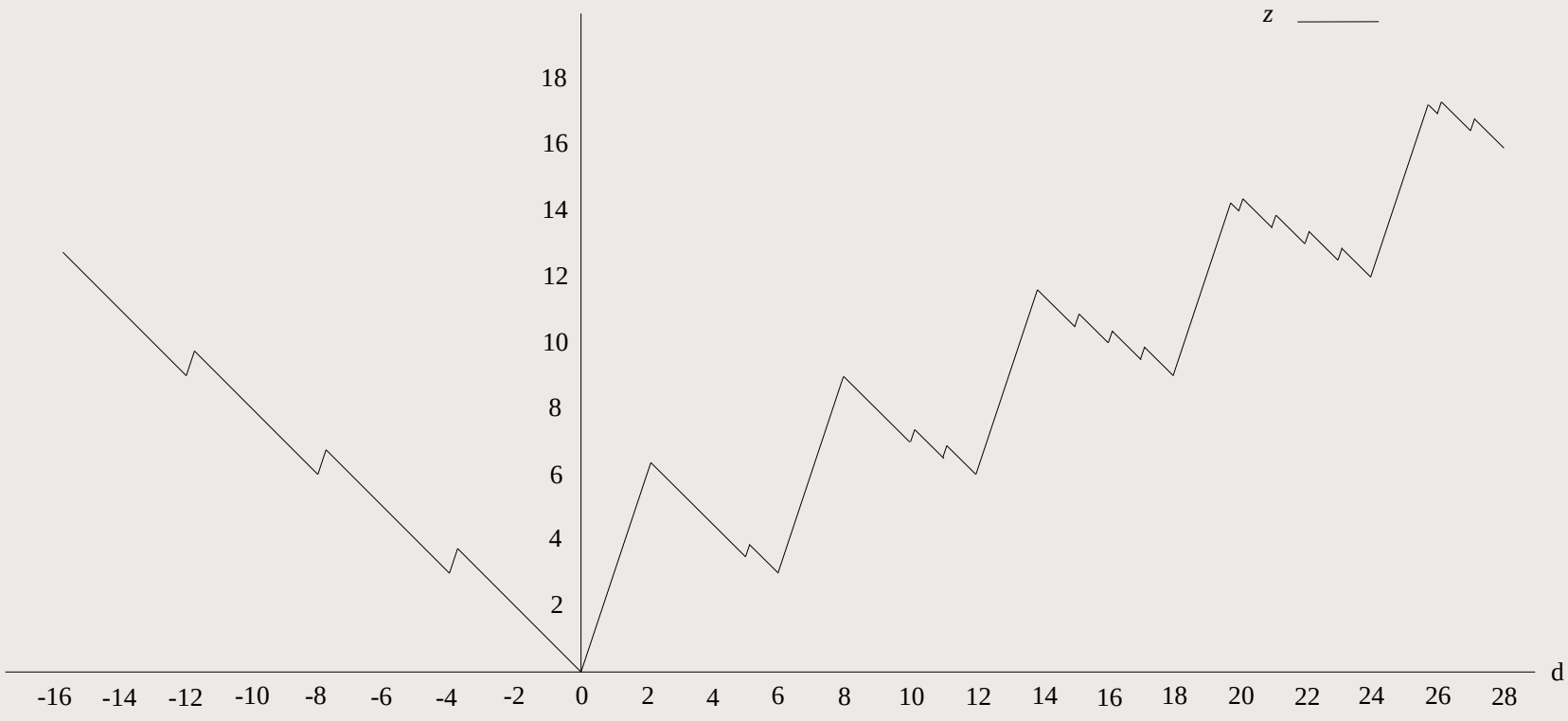
Theorem 4
Let x^* be an optimal solution to the primal problem with right-hand side b. Define the function f as

$$f(d) = c_I x_I^* + z_C(d - A_I x_I^*) \quad \forall d \in \mathbb{R}^m.$$

Then, $f(d) \geq z(d) \ \ \forall d \in \mathbb{R}^m$ with $f(b) = z(b)$, and hence, is a strong upper bounding function at b.

Example

$$\begin{array}{rl} \min & 3x_1 + \frac{7}{2}x_2 + 3x_3 + 6x_4 + 7x_5 + 5x_6 \\ \text{s.t} & 6x_1 + 5x_2 - 4x_3 + 2x_4 - 7x_5 + x_6 = b \quad \text{and} \\ & x_1, x_2, x_3 \in \mathbb{Z}_+, x_4, x_5, x_6 \in \mathbb{R}_+. \end{array} \tag{SP}$$



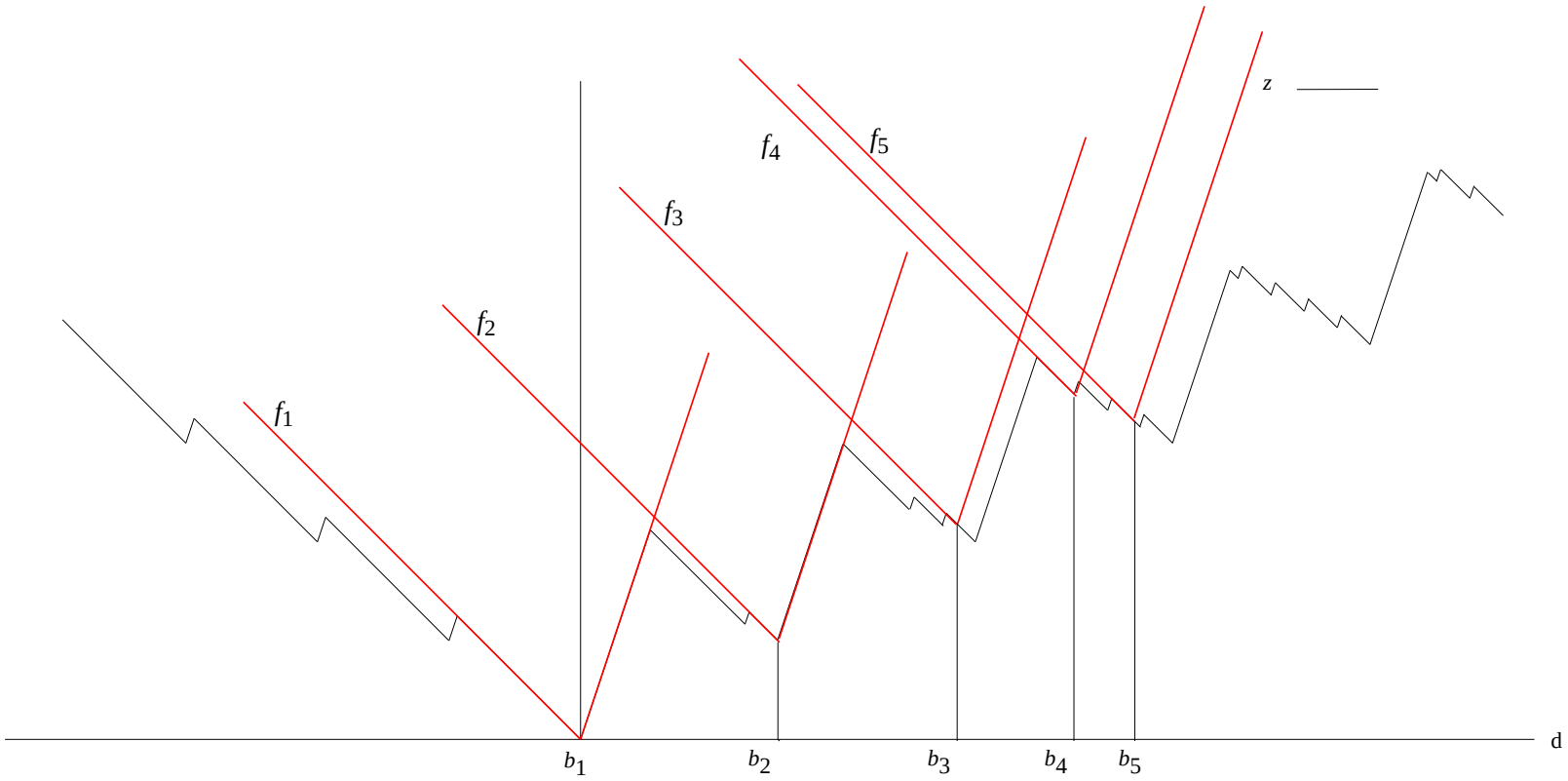


Figure: Upper bounding functions obtained at right-hand sides $b_i, i = 1, \ldots, 5$.

Aggregated Upper Bounding Approximations

Let $\mathcal{F}^U$ be the set of upper bounding functions obtained for each $b \in U \subset \mathbb{R}^m$ where Ui s some collection of right-hand sides. Then

$$F(d) = \min_{f \in \mathcal{F}^U} \{f(d)\} \quad \forall d \in \mathbb{R}^m$$

would be a global upper approximation of the value function.

A Branch and Cut Algorithm

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

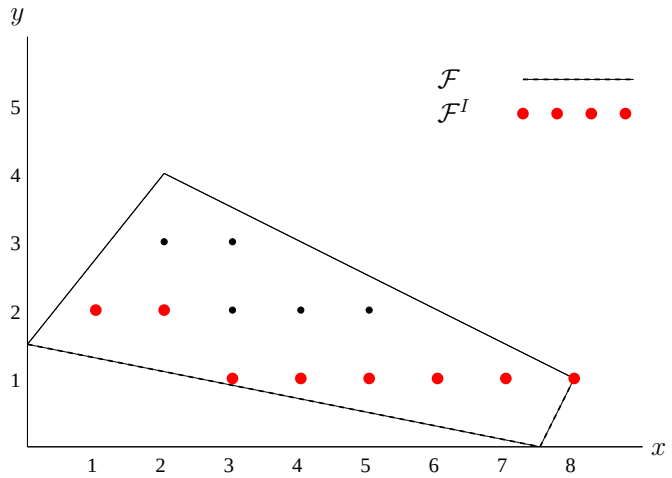
Components

- Bounding methods ($\Leftarrow$ this talk)
- Branching methods ($\Leftarrow$ this talk)
- Search strategies
- Preprocessing methods
- Primal heuristics

- In the remainder of the talk, we address development of these components.

Lower Bounds

Lower bounds can be obtained by relaxing the value function constraint.



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

where $\bar{z}_{LL}$ is an *upper approximation* of the value function of the lower-level problem.

- The upper approximation $\bar{z}_{LL}$ is assumed to be piecewise linear and generated dynamically.
- If the number of pieces is "small," we can reformulate the above in the usual way using integer variables.

Bilevel Feasibility Check

- Let $(\hat{x}, \hat{y})$ be a solution to the lower bounding problem.
- We fix $x = \hat{x}$ and solve the lower-level problem

$$\min_{y \in \mathcal{P}_L^I(\hat{x})} d^2 y \tag{35}$$

  with the fixed upper-level solution $\hat{x}$.
- Let y^* be the solution to (35).
  - $(\hat{x}, y^*)$ is bilevel feasible $\Rightarrow c^1\hat{x} + d^1 y^*$ is a valid upper 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 so that $(\hat{x}, \hat{y})$ is no longer feasible.
  - Branch on a disjunction violated by $(\hat{x}, \hat{y})$.

Bilevel Feasibility Cut (Pure Integer Case)

Let

$$A := \begin{bmatrix} A^1 \\ A^2 \end{bmatrix}, \quad G := \begin{bmatrix} 0 \\ G^2 \end{bmatrix}, \quad \text{and} \quad b := \begin{bmatrix} b^1 \\ b^2 \end{bmatrix}.$$

A basic feasible solution $(\hat{x}, \hat{y}) \in \Omega^I$ to the lower 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

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

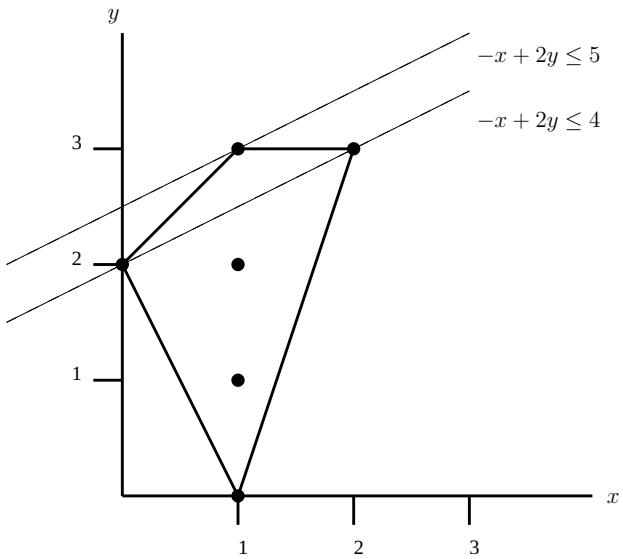
and $\sum_{i \in I} a_i' x + g_i' y \leq \sum_{i \in I} b_i$ is valid for Ω.

Bilevel Feasibility Cut (cont.)

A Valid Inequality

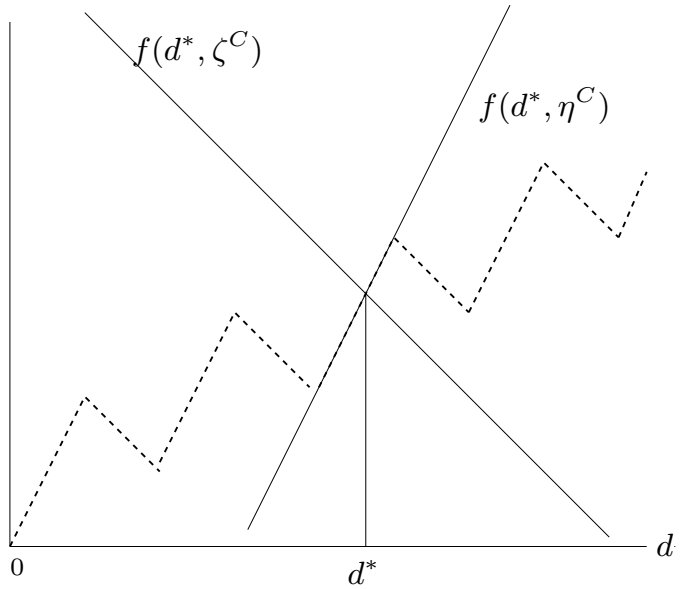
$\sum_{i \in I} a_i' x + g_i' y \leq \sum_{i \in I} b_i - 1$ is valid for $\Omega^I \setminus \{(x, y)\}$.

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



This yields a finite algorithm in the pure integer case.

Value Function Disjunction (Single Constraint Case)



For any $d \leq d^*$,

$$z(d) \leq \max\{f(d^*, \zeta^C), f(d^*, \eta^C)\} = f(d^*, \zeta^C).$$

Similarly, for any $d \geq d^*$,

$$z(d) \leq \max\{f(d^*, \zeta^C), f(d^*, \eta^C)\} = f(d^*, \eta^C).$$

Value Function Disjunction (cont.)

Thus, we have the following disjunction.

Bilevel Feasibility Disjunction

$$b^2 - A^2x \leq b^2 - A^2\hat{x} \quad \text{AND} \quad d^2y \leq f(b^2 - A^2\hat{x}, \zeta^C)$$

$$\text{OR}$$

$$b^2 - A^2x \geq b^2 - A^2\hat{x} \quad \text{AND} \quad d^2y \leq f(b^2 - A^2\hat{x}, \eta^C).$$

Such a disjunction can be used to either branch or cut when solutions $(\hat{x}, \hat{y}) \in \Omega^I$ such that $\hat{y} \notin M^I(\hat{x})$ are found.

Algorithms Based on Lower Approximation

- It is more difficult to develop algorithms based on lower approximations in the general case.
- The main challenge is that we need to know the lower level solution at each step.
- This is different than in the case of recourse problems.
- We are confident that it is possible to overcome these challenges, but these ideas are preliminary.

Implementation

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 perform the branch and bound.
- The COIN Branch and Cut (CBC) framework for solving the MILPs.
- 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 CBC.
- The Open Solver Interface (OSI) for interfacing with CBC and CLP.

What Is Implemented

MibS is still in its infancy and is not fully general. Currently, we have:

- Bilevel feasibility cuts (pure integer case).
- Specialized methods (primarily cuts) for pure binary at the upper level.
- Specialized methods for interdiction problems.
- Disjunctive cuts based on the value function for lower-level problems with a single constraint.
- Several primal heuristics.
- Simple preprocessing.

Preliminary Results from Knapsack Interdiction

	Maximum Infeasibility			Strong Branching		
$2n$	Avg Nodes	Avg Depth	Avg CPU (s)	Avg Nodes	Avg Depth	Avg CPU (s)
20	359.30	8.65	9.32	358.30	8.65	11.07
22	658.40	9.85	18.50	658.20	9.85	18.92
24	1414.80	10.85	46.03	1410.80	10.75	46.46
26	2725.00	12.05	97.55	2723.50	12.05	100.17
28	5326.40	12.90	214.97	5328.60	12.95	220.26
30	10625.00	14.05	482.70	10638.00	14.10	538.32

- Interdiction problems in which the lower-level problems are binary knapsack problems.
- Data was taken from the *Multiple Criteria Decision Making* library and modified to suit our setting.
- Results for each problem size reflect the average of 20 instances.
- These instances were running using the interdiction customization.

Preliminary Results from Assignment Interdiction

Instance	Nodes	Depth	CPU (s)
2AP05-1	6203	33	290.25
2AP05-2	3881	32	384.97
2AP05-3	3909	32	205.93
2AP05-4	2441	36	102.66
2AP05-5	3505	33	119.18
2AP05-6	2031	35	80.31
2AP05-7	2957	29	153.02
2AP05-8	3549	32	224.77
2AP05-9	2271	33	111.13
2AP05-10	3299	31	211.07
2AP05-11	707	33	35.13
2AP05-12	407	18	29.51
2AP05-13	391	18	23.80
2AP05-14	3173	28	261.08
2AP05-15	2509	32	127.05
2AP05-16	1699	29	44.61
2AP05-17	5417	29	201.34
2AP05-18	5785	32	176.67
2AP05-19	2259	32	79.70
2AP05-20	2585	31	77.35
2AP05-21	6039	33	161.44
2AP05-22	2479	29	48.06
2AP05-23	1519	25	49.40
2AP05-24	15	5	1.32
2AP05-25	3857	31	115.97

- Here, the lower-level problems are binary assignment problems.
- Data also taken from *Multiple Criteria Decision Making* library.
- Problems have 50 variables and 45 constraints.

Conclusions and Future Work

- Preliminary testing to date has revealed that these problems can be extremely difficult to solve in practice.
- What we have implemented so far has only scratched the surface.
- Currently, we are focusing on special cases where we can get traction.
  - Interdiction problems
  - Stochastic integer programs
- Much work remains to be done.
- Please join us!

References I