Integer Programming
ISE 418
Lecture 12

Dr. Ted Ralphs
Reading for This Lecture

- Nemhauser and Wolsey Sections II.1.1-II.1.3, II.1.6, II.4.3
- Wolsey Chapter 8
- CCZ Chapter 3, Section 7.5
Describing $\text{conv}(\mathcal{S})$

- We have seen that, in theory, $\text{conv}(\mathcal{S})$ has a finite description.
- If we “simply” construct that description, we could turn our MILP into an LP.
- So why aren’t IPs easy to solve?
  - The size of the description is generally HUGE!
  - The number of facets of the TSP polytope for an instance with 120 nodes is more than $10^{100}$ times the number of atoms in the universe.
  - It is physically impossible to write down a description of this polytope.
  - Not only that, but it is very difficult in general to generate these facets (this problem is not polynomially solvable in general).
For Example

• For a TSP of size 15
  – The number of subtour elimination constraints is 16,368.
  – The number of comb inequalities is 1,993,711,339,620.
  – These are only two of the know classes of facets for the TSP.

• For a TSP of size 120
  – The number of subtour elimination constraints is 0.6 \times 10^{36}!
  – The number of comb inequalities is approximately 2 \times 10^{179}!
Valid Inequalities Revisited

- Recall that the inequality denoted by \((\pi, \pi_0)\) is valid for a polyhedron \(Q\) if \(\pi x \leq \pi_0 \ \forall x \in Q\).
- Note that an inequality \((\pi, \pi_0)\) is valid if and only if
  \[
  \pi_0 \geq \max_{x \in Q} \pi^\top x
  \]
- Alternatively, an inequality \((\pi, \pi_0)\) is valid if
  \[
  \pi_0 \geq F(b),
  \]
  where \(F\) is a dual function with respect to the optimization problem
  \[
  \max_{x \in Q} \pi^\top x
  \]
- In fact, many classes of valid inequalities used in solvers are generated in this way.
- Thus, there is an inextricable link between valid inequalities and optimization.
Cutting Planes

- The term *cutting plane* usually refers to an inequality valid for $\text{conv}(S)$, but which is violated by the solution to the (current) LP relaxation.

- Cutting plane methods attempt to improve the bound produced by the LP relaxation by iteratively adding cutting planes to the initial LP relaxation.

- Taken to its limit, this is an algorithm for solving MILPs that fits into the general “dual improvement” framework.

- Adding such inequalities to the LP relaxation *may* improve the bound (this is not a guarantee).
**The Separation Problem**

- Formally, the problem of generating a cutting plane can be stated as follows.

  **Separation Problem**: Given a polyhedron $Q \subseteq \mathbb{R}^n$ and $x^* \in \mathbb{R}^n$, determine whether $x^* \in Q$ and if not, determine $(\pi, \pi_0)$, an inequality valid for $Q$ such that $\pi x^* > \pi_0$.

- This problem is stated here independent of any solution algorithm.

- However, it is typically used as a subroutine inside an iterative method for improving the LP relaxation.

- In such a case, $x^*$ is the solution to the LP relaxation (of the current formulation, including previously generated cuts).

- We will see that the difficulty of solving this problem exactly is strongly tied to the difficulty of the optimization problem itself.

- Any algorithm for solving the separation problem can be immediately leveraged to produce an algorithm for solving the optimization problem.

- This algorithm is known as the **cutting plane algorithm**.
Generic Cutting Plane Method

Let \( \mathcal{P} = \{ x \in \mathbb{R}^n \mid Ax \leq b \} \) be the initial formulation for

\[
\max \{ c^\top x \mid x \in \mathcal{S} \}, \tag{MILP}
\]

where \( \mathcal{S} = \mathcal{P} \cap \mathbb{Z}_+^r \times \mathbb{R}_+^{n-p} \), as defined previously.

Cutting Plane Method

\[
\begin{align*}
P_0 &\leftarrow \mathcal{P} \\
k &\leftarrow 0 \\
\textbf{while} \; \text{TRUE} \; \textbf{do} \\
&\quad \text{Solve the LP relaxation } \max \{ c^\top x \mid x \in \mathcal{P}_k \} \text{ to obtain a solution } x^k \\
&\quad \text{Solve the problem of separating } x^k \text{ from } \text{conv}(\mathcal{S}) \\
&\quad \textbf{if } x^k \in \text{conv}(\mathcal{S}) \; \textbf{then} \\
&\quad &\quad \text{STOP} \\
&\quad \textbf{else} \\
&\quad &\quad \text{Determine an inequality } (\pi^k, \pi^k_0) \text{ valid for } \text{conv}(\mathcal{S}) \text{ but for which } \\
&\quad &\quad &\pi^\top x^k > \pi^k_0. \\
&\quad \textbf{end if} \\
&\quad \mathcal{P}_{k+1} \leftarrow \mathcal{P}_k \cap \{ x \in \mathbb{R}^n \mid (\pi^k)^\top x \leq \pi^k_0 \} \\
&\quad k \leftarrow k + 1 \\
\textbf{end while}
\]
Questions to be Answered

• How do we solve the separation problem in practice?
• Will this algorithm terminate?
• If it does terminate, are we guaranteed to obtain an optimal solution?
The Separation Problem as an Optimization Problem

**Separation Problem**: Given a polyhedron $Q \subseteq \mathbb{R}^n$ and $x^* \in \mathbb{R}^n$, determine whether $x^* \in Q$ and if not, determine $(\pi, \pi_0)$, a valid inequality for $Q$ such that $\pi x^* > \pi_0$.

- Closer examination of the separation problem for a polyhedron reveals that it is in fact an optimization problem.
- Consider a polyhedron $Q \subseteq \mathbb{R}^n$ and $x^* \in \mathbb{R}^n$.
- The separation problem can be formulated as

$$\max\{\pi x^* - \pi_0 \mid \pi^T x \leq \pi_0 \ \forall x \in P, (\pi, \pi_0) \in \mathbb{R}^{n+1}\} \quad \text{(SEP)}$$

along with some normalization to prevent (SEP) being unbounded.
- When $Q$ is a polytope, we can reformulate this problem as the LP

$$\max\{\pi x^* - \pi_0 \mid \pi^T x \leq \pi_0 \ \forall x \in E\},$$

where $E$ is the set of extreme points of $Q$.
- When $Q$ is not bounded, the reformulation must account for the extreme rays of $Q$. 
The Normalization

• There are multiple ways to normalize, e.g.,
  
  – \( \pi_0 = 1 \) or
  – \( \|\pi\| = 1 \).

• These are equivalent with respect to reducing the separation problem to an optimization problem

• Different normalizations will, however, result in different optimal solutions and will behave differently in a computational setting.

• The issue of how to normalize will come up again in later lectures.
The Polar

Definition 1. The polar of a set $S$ is $S^* = \{y \in \mathbb{R}^n \mid yx \leq 1 \ \forall x \in S\}$.

Theorem 1. Given $a^1, \ldots, a^m \in \mathbb{Q}^n$ and $0 \leq k \leq m$, let

\[
Q_1 = \{x \in \mathbb{R}^n \mid a^i x \leq 1, \ i = 1, \ldots, k; a^i x \leq 0, \ i = k + 1, \ldots, m\}
\]
\[
Q_2 = \text{conv}(\{0, a^1, \ldots, a^k\}) + \text{cone}(\{a^{k+1}, \ldots a^n\})
\]

Then $Q_1^* = Q_2$ and $Q_2^* = Q_1$

- From this definition, we can see that if $Q$ is a polyhedron containing the origin, then have that

1. $Q^*$ is also a polyhedron containing the origin;
2. $Q^{**} = Q$;
3. $Q^*$ is bounded if and only if $Q$ contains the origin in its interior;
4. $\text{aff}(Q^*)$ is the orthogonal complement of $\text{lin}(Q)$ and $\dim(Q^*) + \dim(\text{lin}(Q)) = n$. 
Interpreting the Polar

• The polar can be roughly interpreted as the (normalized) set of all valid inequalities.

• Without some normalization, it would contain all scalar multiples of each inequality.

• Because of the normalization used here, the polar is sometimes called the 1-Polar in this context.

• There is a one-to-one correspondence between the facets of the polyhedron and the extreme points of the 1-Polar when
  – the polyhedron is full-dimensional and
  – the origin is in its interior,

• Hence, the separation problem can be seen as an optimization problem over the polar.
The Membership Problem

**Membership Problem:** Given a polyhedron $Q \subseteq \mathbb{R}^n$ and $x^* \in \mathbb{R}^n$, determine whether $x^* \in Q$.

- The membership problem is a decision problem and is closely related to the separation problem.
- In fact, the dual of (SEP) is a formulation for the membership problem:

$$
\min_{\lambda \in \mathbb{R}^E_+} \left\{ 0^\top \lambda \mid E\lambda = x^*, 1^\top \lambda = 1 \right\}, \tag{MEM}
$$

where $E$ is a matrix whose columns are the extreme points of $Q$.

- In other words, we try to express $x^*$ as a convex combination of extreme points of $Q$.
- When this LP is infeasible, the certificate is a separating hyperplane.
- We can solve this LP by column generation.
- In each iteration, a new column is “generated” by optimizing over $Q$.
- We can picture this algorithm in the “primal space” to understand what it’s doing.
Example: Separation Algorithm with Optimization Oracle

Figure 1: Polyhedron and point to be separated
Example: Separation Algorithm with Optimization Oracle

Figure 2: Iteration 1
Example: Separation Algorithm with Optimization Oracle

Figure 3: Iteration 2
Example: Separation Algorithm with Optimization Oracle

Figure 4: Iteration 3
Example: Separation Algorithm with Optimization Oracle

Figure 5: Iteration 4
Example: Separation Algorithm with Optimization Oracle

Figure 6: Iteration 5
The Separation Problem for the 1-Polar

• The column generation algorithm for solving (MEM) can be interpreted as a cutting plane algorithm for solving (SEP).

• The separation problem (SEP) for $Q$ has one inequality for each extreme point of $Q$.

• We can generate these inequalities using a cutting plane algorithm.

• This is a bit circular...this requires solving the separation problem for $Q^*$, the 1-Polar.

• For a given $\pi^* \in \mathbb{R}^n$, the separation problem for $Q^*$ is to determine whether $\pi^* \in Q^*$ and if not, determine $x \in E$ such that $\pi^* x < 1$.

• In other words, we are asking whether $\pi^*$ is a valid inequality for $Q$.

• As before, this problem can be formulated as

$$\max\{\pi^* x \mid x \in Q\},$$

which is an optimization problem over $Q$!
Formal Equivalence of Separation and Optimization

**Separation Problem:** Given a polyhedron $Q \subseteq \mathbb{R}^n$ and $x^* \in \mathbb{R}^n$, determine whether $x^* \in P$ and if not, determine $(\pi, \pi_0)$, a valid inequality for $Q$ such that $\pi x^* > \pi_0$.

**Optimization Problem:** Given a polyhedron $Q$, and a cost vector $c \in \mathbb{R}^n$, determine $x^*$ such that $cx^* = \max\{cx : x \in Q\}$.

**Theorem 2.** For a family of rational polyhedra $Q(n, T)$ whose input length is polynomial in $n$ and $\log T$, there is a polynomial-time reduction of the linear programming problem over the family to the separation problem over the family. Conversely, there is a polynomial-time reduction of the separation problem to the linear programming problem.

- The parameter $n$ represents the dimension of the space.
- The parameter $T$ represents the largest numerator or denominator of any coordinate of an extreme point of $Q$ (the vertex complexity).
- The *ellipsoid algorithm* provides the reduction of linear programming separation to separation.
- *Polarity* provides the other direction.
Proof: The Ellipsoid Algorithm

• The ellipsoid algorithm is an algorithm for solving linear programs.

• The implementation requires a subroutine for solving the separation problem over the feasible region (see next slide).

• We will not go through the details of the ellipsoid algorithm.

• However, its existence is very important to our study of integer programming.

• Each step of the ellipsoid algorithm, except that of finding a violated inequality, is polynomial in

  – $n$, the dimension of the space,
  – $\log T$, where $T$ is the largest numerator or denominator of any coordinate of an extreme point of $Q$, and
  – $\log \|c\|$, where $c \in \mathbb{R}^n$ is the given cost vector.

• The entire algorithm is polynomial if and only if the separation problem is polynomial.
Classes of Inequalities

- As we have just shown, producing general facets of $\text{conv}(S)$ is as hard as optimizing over $S$.

- Thus, the approach often taken is to solve a “relaxation” of the separation problem.

- This “relaxation” is usually obtained in one of several ways.
  - It can be obtained in the usual way by relaxing some constraints to obtain a more tractable problem.
  - The “structure” of the inequalities may be somehow restricted to make the right-hand side easy to compute.
  - We may also use a dual function to compute the right-hand side rather than computing the “optimal” right-hand side.

- We will see examples of all these in later lectures.

- In either of the first two cases, the class of inequalities we want to generate typically defines a polyhedron $C$.

- $C$ is what we earlier called the closure.

- The separation problem for the class is the separation problem over the closure.