Integer Programming
ISE 418

Lecture 29

Dr. Ted Ralphs

Reading for This Lecture

- Nemhauser and Wolsey Sections I.6.1, III.1.1-III.1.3
- Wolsey Chapter 3
- CCZ Chapter 4

When is an IP Easy to Solve?

- We will consider a particular class of MILPs to be "easy" when we can solve all instances in the class in polynomial time.
- We will see that there are a number of properties that indicate an IP is easy:
  1. Existence of an efficient optimization algorithm,
  2. Existence of an efficient separation algorithm for the $\text{conv}(\mathcal{S})$.
  3. Existence of a complete description of $\text{conv}(\mathcal{S})$ of polynomial size,
  4. Existence of a short certificate of optimality, or
  5. Existence of an efficiently solvable strong dual problem.
- We will see that under certain conditions, Properties 1 and 2 are equivalent.
- Property 3 is, in some sense, the strongest—it implies all other properties.

Polynomial Equivalence of Separation and Optimization

Separation Problem: Given a polyhedron $\mathcal{P} \subseteq \mathbb{R}^n$ and $x^* \in \mathbb{R}^n$, determine whether $x^* \in \mathcal{P}$ and if not, determine (π, π_0), a valid inequality for $\mathcal{P}$ such that $\pi x^* > \pi_0$.

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

Theorem 1. *For a family of rational polyhedra $\mathcal{P}(n, T)$ 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 $\mathcal{P}$ (the *vertex complexity*).
- The *ellipsoid algorithm* provides the reduction of linear programming separation to separation.
- *Polarity* provides the other direction.

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 is the largest numerator or denominator of any coordinate of an extreme point of $\mathcal{P}$, 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.

The Membership Problem

- The *membership problem* is to determine whether $x^* \in \mathcal{P}$, for $x^* in \mathbb{R}^n$ and a polyhedron $\mathcal{P}$.
- The membership problem is a decision problem and is closely related to the separation problem.
- Consider the following approach to solving the membership problem.
  - We try to express x^* as a convex combination of extreme points of $\mathcal{P}$.
  - This problem can be formulated as a linear program with a column for each extreme point.
  - If this linear program is infeasible, the certificate is a separating hyperplane.
  - This linear program can be solved by column generation.
  - Note that the column generation subproblem is the separation problem in the dual.
  - Thus, we can solve this linear program in polynomial time if and only if we can optimize over $\mathcal{P}$.

Example: Minimum Weight s–t Cut

- Consider the problem of finding a minimum weight $s - t$ cut in a graph $G = (V, E)$ with edge weights $c \in \mathbb{R}^E$.
- One formulation of this problem as a linear program is

$$\begin{aligned} \min \sum_{e \in E} c_e y_e & \\ \sum_{e \in K} y_e \geq 1 & \quad \forall K \in \mathcal{K} \\ 0 \leq y_e \leq 1 & \quad \forall e \in E \end{aligned}$$

  where $\mathcal{K}$ is the family of $s - t$ paths in G.

- Questions:
  - Can we solve this linear program efficiently?
  - Will the solution to the linear program be integral?
- The first question above amounts to whether we can solve the separation problem efficiently.
- Given a $y^* \in \mathbb{R}^E$ satisfying the bound constraints, can we determine efficiently whether it satisfies the remaining constraints?

Example: Minimum Weight s–t Cut (cont.)

- We already know that the minimum cut problem is polynomially solvable.
- However, this formulation of the problem is not of polynomial size.
- Since the separation problem is equivalent to the shortest path problem, we can conclude that the linear program is polynomially solvable.
- The question still remains whether the solution to this linear program will be integral.

Integral Polyhedra

- The theory of integral polyhedra in this lecture applies primarily in the context of pure integer programs.
- In this setting, an *integral point* is just a member of $\mathbb{Z}^n$.

  Definition 1. *A nonempty polyhedron* $\mathcal{P}$ *is said to be* integral *if each of its nonempty faces contains an integral point.*

  Proposition 1. *A nonempty polyhedron* $\mathcal{P} = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ *with* $rank(A) = n$ *is integral if and only if all of its extreme points are integral.*

- We will assume for the remainder of the section on integral polyhedra that all nonempty polyhedra have extreme points.
- Why do we care about integral polyhedra?

Integral Polyhedra

Consider the linear programming problem $z_{LP} = \max\{cx \mid x \in \mathcal{P}\}$ for a given polyhedron $\mathcal{P}$.

Proposition 2. *The following statements are equivalent:*

1. $\mathcal{P}$ *is integral*
2. *The associated LP has an integral optimal solution for all* $c \in \mathbb{R}^n$ *for which an optimal solution exists.*
3. *The associated LP has an integral optimal solution for all* $c \in \mathbb{Z}^n$ *for which an optimal solution exists.*
4. z_{LP} *is integral for all* $c \in \mathbb{Z}^n$ *for which an optimal solution exists.*

If a polyhedron is integral, then we can optimize over it using linear programming techniques.

Total Dual Integrality

Definition 2. *A system of linear inequalities* $Ax \leq b$ *is called* totally dual integral *(TDI) if, for all* $c \in \mathbb{Z}^n$ *such that* $z_{LP} = \max\{cx \mid Ax \leq b\}$ *is finite, the dual* $\min\{yb \mid yA = c, y \in \mathbb{R}^m_+\}$ *has an integral optimal solution.*

- Note that this definition does not pertain to polyhedra, but to systems of inequalities.
- The importance of this definition is that if $Ax \leq b$ is TDI and b is integral, then $\mathcal{P} = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ must be integral (why?).
- Note that the property of being TDI is sensitive to scaling.
- Every polyhedron has a representation that is TDI.
- In fact, a polyhedron is integral *if and only if* it has a TDI representation where the right-hand side is integral.

Total Unimodularity

Definition 3. *An $m \times n$ integral matrix A is* totally unimodular *(TU) if the determinant of every square submatrix is 0, 1, or -1.*

- Obviously, only matrices with entries of 0, 1, and -1 can be TU.
- If A is TU, then $\mathcal{P}(b) = \{x \in \mathbb{R}^n_+ \mid Ax \leq b\}$ is integral *for all* $b \in \mathbb{Z}^m$.
- How could we go about proving this?
- TU is a very strong property.
- If the constraint matrix of an integer program is TU, then it can be solved using linear programming techniques.

Properties of Totally Unimodular Matrices

The following are equivalent:

1. A is TU.
2. The transpose of A is TU.
3. (A, I) is TU.
4. A matrix obtained by deleting a unit row/column from A is TU.
5. A matrix obtained by multiplying a row/column of A by -1 is TU.
6. A matrix obtained by interchanging two rows/columns of A is TU.
7. A matrix obtained by duplicating rows/columns of A is TU.
8. A matrix obtained by a pivot operation on A is TU.

- We can easily show that if A is TU, it remains so after adding slack variables, adding simple bounds on the variables, or adding ranges on the constraints (how?).
- We can also show that the polyhedron corresponding to the dual LP is integral.

The Converse

- We have just seen that if the constraint matrix is TU, then the polyhedron is integral.
- In fact, the converse is true too!

  Proposition 3. *If* $\mathcal{P}(b) = \{x \in \mathbb{R}^n_+ \mid Ax \leq b\}$ *is integral for all* $b \in \mathbb{Z}^m$*, then* A *is TU.*

Recognizing Totally Unimodular Matrices

- At this point, it appears difficult to recognize TU matrices.
- However, we have a characterization that will be useful.

Proposition 4. *A is TU if and only if for every $J \subseteq \{1, \ldots, n\}$, there exists a partition J_1, J_2 of J such that*

$$\left| \sum_{j \in J_1} a_{ij} - \sum_{j \in J_2} a_{ij} \right| \leq 1 \text{ for } i = 1, \ldots, m.$$

Corollary 1. *If the (0, 1, -1) matrix A has no more than two nonzero entries in each column, and if $\sum_i a_{ij} = 0$ if column j contains two nonzero coefficients, then A is TU.*

Examples of TU Matrices

- It follows easily from the corollary that the node-arc incidence matrix of a directed graph is a TU matrix.
- This leads to easy proofs of integral min-max results such as the max flow-min cut theorem.
- Another example of a TU matrix is the node-edge incidence matrix of a bipartite graph.

  Definition 4. *A (0, 1) matrix* A *is called an* interval matrix *if in each column, the 1's appear consecutively.*

- Interval matrices are also TU.
- It is interesting to note that any integer program with a (0, 1) constraint matrix has a relaxation defined by an interval matrix (see page 545 of Nemhauser and Wolsey).

Network Matrices

- A *network matrix* is obtained from a node-arc incidence matrix of a graph after deleting one (dependent) row and performing any number of simplex pivots.
- In other words, it is any matrix that could appear as a tableau when solving a minimum cost network flow problem.
- It is easy to see that all network matrices are TU.
- More surprising is the fact that "nearly all" TU matrices are network matrices!

The TU Recognition Problem

Proposition 5. *Every TU matrix that is not a network matrix or one of the two matrices below can be constructed from these matrices using the rules of the Propositions 2.1 and 2.11 from Nemhauser and Wolsey.*

$$\begin{pmatrix} 1 & -1 & 0 & 0 & -1 \\ -1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

- This observation tells us that the TU recognition problem is in NP. What is the certificate?
- In fact, the TU recognition problem is polynomially solvable.