IE418: Integer Programming

Jeff Linderoth

Department of Industrial and Systems Engineering
Lehigh University

24th January 2005

Review

- Name two reasons why anyone would want to “program with integers”?
- What is the knapsack problem?
- (MIP): \( \max \{ c^T x + h^T y \mid Ax + Gy \leq b, x \in \mathbb{Z}^n_+, y \in \mathbb{R}^p_+ \} \)
  - What is Mixed 0-1 Programming?
  - What is Pure Integer Programming
  - What is Binary Programming?
  - What is 0-1 Programming?
- What is \( \mathbb{Q}^n_+ \)?
- What is my son’s name?
Outline

- Combinatorial Optimization Problems
  - Set \{Packing, Covering, Partitioning\}
  - TSP
- Special Ordered Sets
  - Models of Choice—"Pick one!"
  - Piecewise Linear Functions
- Algorithmic Modeling
  - The Bag of Tricks
  - A couple examples

Selecting from a Set

- We can use constraints of the form \( \sum_{j \in T} x_j \geq 1 \) to represent that at least one item should be chosen from a set \( T \).
  - Similarly, we can also model that at most one or exactly one item should be chosen.
- Example: Set covering problem
- If \( A \) in a 0-1 matrix, then a set covering problem is any problem of the form

\[
\begin{align*}
\min & \quad e^T x \\
\text{s.t.} & \quad Ax \geq e^1 \\
& \quad x_j \in \{0, 1\} \quad \forall j
\end{align*}
\]

- Set Packing: \( Ax \leq e \)
- Set Partitioning: \( Ax = e \)

\(^1\)It is common to denote the vector of 1's as \( e \).
A combinatorial optimization problem \( CP = (N, \mathcal{F}) \) consists of:
- A finite ground set \( N \),
- A set \( \mathcal{F} \subseteq 2^N \) of feasible solutions, and
- A cost function \( c \)

Each row of \( A \) represents an item from \( N \)

Each column \( A_j \) represents a subset \( N_j \in \mathcal{F} \) of the items.

Each variable \( x_j \) represents selecting subset \( N_j \).

The constraints (for covering and partitioning) say that
\[ \bigcup \{ j \mid x_j = 1 \} N_j = N. \]

In other words, each item must appear in at least, (or exactly one, selected subset.

Vehicle Routing

<table>
<thead>
<tr>
<th></th>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( x_3 )</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer 1:</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
</tr>
<tr>
<td>Customer 2:</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>\ldots</td>
</tr>
<tr>
<td>Customer 3:</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>\ldots</td>
</tr>
<tr>
<td>Customer 4:</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>\ldots</td>
</tr>
<tr>
<td>Customer 5:</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>\ldots</td>
</tr>
</tbody>
</table>

This is a very flexible modeling trick

You can list all feasible routes, allowing you to handle “weird” constraints like time windows, strange precedence rules, nonlinear cost functions, etc.
The Farmer’s Daughter

- This is *The Most Famous Problem in Combinatorial Optimization!*
- A traveling salesman must visit all his cities at minimum cost.
- Given directed (complete) graph with node set \( N \).
  \( G = (N, N \times N) \)
- Given costs \( c_{ij} \) of traveling from city \( i \) to city \( j \)
- Find a minimum cost *Hamiltonian Cycle* in \( G \)
- **Variables:** \( x_{ij} = 1 \) if and only if salesman goes from city \( i \) to city \( j \)

**TSP (cont.)**

\[
\begin{align*}
\text{min} & \quad \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} \\
\sum_{i \in N} x_{ij} &= 1 \quad \forall j \in N \quad \text{Enter Each City} \\
\sum_{j \in N} x_{ij} &= 1 \quad \forall i \in N \quad \text{Leave Each City} \\
x_{ij} &\in \{0, 1\} \quad \forall i \in N, \forall j \in N
\end{align*}
\]

Subtour elimination constraint:

\[
\sum_{i \in S} \sum_{j \in N \setminus S} x_{ij} \geq 1 \quad \forall S \subseteq N, 2 \leq |S| \leq |N| - 2
\]

Alternatively:

\[
\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1 \quad \forall S \subseteq N, 2 \leq |S| \leq |N| - 2
\]
TSP Trivia Time!

What is This Number?

10185179881672430431342284204689080525734196832968125318070224677190649881668353091698688.

- Is this...
  - a) The number of times an undergraduate student asked me where room 355 was this week?
  - b) The number of subatomic particles in the universe?
  - c) The number of subtour elimination constraints when $|N| = 299$.
  - d) All of the above?
  - e) None of the above?

Answer Time

- The answer is (e). (a)–(c) are all too small (as far as I know) :-

  - “Exponential” is really big.

  - Yet people have solved TSP’s with $|N| > 16,000$!

  - You will learn how to solve these problems too!

  - The “trick” is to only add the subset of constraints that are necessary to prove optimality.

    - This is a trick known as branch-and-cut, and you will learn lots about branch-and-cut in this course.
Modeling a Restricted Set of Values

- We may want variable $x$ to only take on values in the set \{a_1, \ldots, a_m\}.
- We introduce $m$ binary variables $y_j$, $j = 1, \ldots, m$ and the constraints

  $$x = \sum_{j=1}^{m} a_j y_j,$$

  $$\sum_{j=1}^{m} y_j = 1,$$

  $$y_j \in \{0, 1\}$$

- The set of variables \{y_1, y_2, \ldots y_m\} is called a special ordered set (SOS) of variables.

Example—Building a warehouse

- Suppose we are modeling a facility location problem in which we must decide on the size of a warehouse to build.
- The choices of sizes and their associated cost are shown below:

<table>
<thead>
<tr>
<th>Size</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>180</td>
</tr>
<tr>
<td>40</td>
<td>320</td>
</tr>
<tr>
<td>60</td>
<td>450</td>
</tr>
<tr>
<td>80</td>
<td>600</td>
</tr>
</tbody>
</table>

Warehouse sizes and costs
Warehouse Modeling

- Using binary decision variables \( x_1, x_2, \ldots, x_5 \), we can model the cost of building the warehouse as
  \[
  \text{COST} \equiv 100x_1 + 180x_2 + 320x_3 + 450x_4 + 600x_5.
  \]

- The warehouse will have size
  \[
  \text{SIZE} \equiv 10x_1 + 20x_2 + 40x_3 + 60x_4 + 80x_5,
  \]

- and we have the SOS constraint
  \[
  x_1 + x_2 + x_3 + x_4 + x_5 = 1.
  \]

Piecewise Linear Cost Functions

- We can use binary variables to model arbitrary piecewise linear functions.

- The function is specified by ordered pairs \((a_i, f(a_i))\)

- We have a binary variable \( y_i \), which indicates whether \( a_i \leq x \leq a_{i+1} \).
Minimizing Piecewise Linear Cost Functions

- To evaluate the function, we will take linear combinations \( \sum_{i=1}^{k} \lambda_i f(a_i) \) of the given functions values.
- This only works if the only two nonzero \( \lambda_i \)'s are the ones corresponding to the endpoints of the interval in which \( x \) lies.

The Key Idea!
If \( y_j = 1 \), then \( \lambda_i = 0 \), \( \forall i \neq j, j+1 \).

A “better” formulation involves the use of special ordered sets of type 2

**SOS2**

- A “better” formulation involves the use of special ordered sets of type 2

\[
\begin{align*}
\min & \quad \sum_{i=1}^{k} \lambda_i f(a_i) \\
\text{s.t.} & \quad \sum_{i=1}^{k} \lambda_i = 1, \\
& \quad \lambda_1 \leq y_1, \\
& \quad \lambda_i \leq y_{i-1} + y_i, \ i = 2, \ldots, k-1, \\
& \quad \lambda_k \leq y_{k-1}, \\
& \quad \sum_{i=1}^{k-1} y_i = 1, \\
& \quad \lambda_i \geq 0, \\
& \quad y_i \in \{0, 1\}.
\end{align*}
\]

- The adjacency conditions of SOS2 are enforced by the solution algorithm
- (All) commercial solvers allow you to specify SOS2
Modeling Disjunctive Constraints

- We are given two constraints \( a^T x \geq b \) and \( c^T x \geq d \) with nonnegative coefficients.
- Instead of insisting both constraints be satisfied, we want at least one of the two constraints to be satisfied.
- To model this, we define a binary variable \( y \) and impose

\[
\begin{align*}
  a^T x & \geq yb, \\
  c^T x & \geq (1 - y)d, \\
  y & \in \{0, 1\}.
\end{align*}
\]

- More generally, we can impose that at least \( k \) out of \( m \) constraints be satisfied with

\[
\begin{align*}
  (a_i')^T x & \geq b_i y_i, & i \in \{1, 2, \ldots m\} \\
  \sum_{i=1}^m y_i & \geq k, \\
  y_i & \in \{0, 1\}.
\end{align*}
\]

The Bag of Tricks

- There are lots of things you can model with binary variables:
  - Indicator variables (Positivity of variables)
    - Limiting the Number of Positive Variables
    - “Fixed Charge” problems
    - Minimum production level
  - Indicator variables (Validity of constraints)
    - Either-or
    - If-then
    - \( k \) out of \( n \)
The Bag of Tricks

- Special Ordered Sets
- Nonconvex regions
- Economies of Scale
- Discrete Capacity Extensions
- Maximax or Minimin

The problem is that sometimes to see the modeling “trick” is difficult. For example...

- Use a 0-1 variable $\delta$ to indicate whether or not the constraint $2x_1 + 3x_2 \leq 1$ is satisfied.
  - $x_1, x_2$ are nonnegative continuous variables that are $\leq 1$
  - $\delta = 1 \iff 2x_1 + 3x_2 \leq 1$

The Slide of Tricks. Indicator Variables...

Definitions

- $\delta$: Indicator variable ($\delta \in \{0, 1\}$).
- $M$: Upper bound on $\sum_{j \in N} a_j x_j - b$
- $m$: Lower bound on $\sum_{j \in N} a_j x_j - b$
- $\epsilon$: Small tolerance beyond which we regard the constraint as having been broken.
  - If $a_j \in \mathbb{Z}$, $x_j \in \mathbb{Z}$, then we can take $\epsilon = 1$. 
Modeling Trick #1

- Indicating Constraint (Non)violation
- Suppose we wish to indicate whether or not an inequality \( \sum_{j \in N} a_j x_j \leq b \) holds by means of an indicator variable \( \delta \).

**Implication We Wish to Model**

\[
\delta = 1 \Rightarrow \sum_{j \in N} a_j x_j \leq b
\]

- This can be represented by the constraint
  - \( \sum_{j \in N} a_j x_j + M \delta \leq M + b \)

**Trick #1... The Logic**

\[
\delta = 1 \Rightarrow \sum_{j \in N} a_j x_j \leq b \iff \sum_{j \in N} a_j x_j + M \delta \leq M + b
\]

- (Thinking)...
  - \( \delta = 1 \Rightarrow \sum_{j \in N} a_j x_j - b \leq 0 \)
  - \( 1 - \delta = 0 \Rightarrow \sum_{j \in N} a_j x_j - b \leq 0 \)
  - \( \sum_{j \in N} a_j x_j - b \leq M(1 - \delta) \)

- Does it work?
  - \( \delta = 0 \Rightarrow \sum_{j \in N} a_j x_j - b \leq M \)
    - (true by definition of \( M \))
  - \( \delta = 1 \Rightarrow \sum_{j \in N} a_j x_j - b \leq 0 \)
### Modeling Trick #2

\[
\sum_{j \in N} a_j x_j \leq b \Rightarrow \delta = 1
\]

- \( \delta = 0 \Rightarrow \sum_{j \in N} a_j x_j \not\leq b \)
- \( \delta = 0 \Rightarrow \sum_{j \in N} a_j x_j > b \)
- \( \delta = 0 \Rightarrow \sum_{j \in N} a_j x_j \geq b + \epsilon \)
  - If \( a_j, x_j \) are integer, we can choose \( \epsilon = 1 \)
  - Model as \( \sum_{j \in N} a_j x_j - (m - \epsilon)\delta \geq b + \epsilon \)
  - \( m \) is a lower bound for the expression \( \sum_{j \in N} a_j x_j - b \)

### Some Last Modeling Tricks

\[
\delta = 1 \Rightarrow \sum_{j \in N} a_j x_j \geq b
\]

- Model as \( \sum_{j \in N} a_j x_j + m\delta \geq m + b \)

\[
\sum_{j \in N} a_j x_j \geq b \Rightarrow \delta = 1
\]

- Model as \( \sum_{j \in N} a_j x_j - (M + \epsilon)\delta \leq b - \epsilon \)
- You can obtain these by just transforming the constraints to \( \leq \) form and using the first two tricks.
Back To Our Example...

- Use a 0-1 variable $\delta$ to indicate whether or not the constraint $2x_1 + 3x_2 \leq 1$ is satisfied.
  - $x_1, x_2$ are nonnegative continuous variables that cannot exceed 1.
  - $\delta = 1 \iff 2x_1 + 3x_2 \leq 1$

- $M$ : Upper Bound on $2x_1 + 3x_2 - 1$. $4$ works
- $m$ : Lower Bound on $2x_1 + 3x_2 - 1$. $-1$ works.
- $\epsilon : 0.1$

Example, Cont.

- (⇒) Recall the trick.
  - $\delta = 1 \Rightarrow \sum_{j \in N} a_j x_j \leq b \iff \sum_{j \in N} a_j x_j + M\delta \leq M + b$
  - $2x_1 + 3x_2 + 4\delta \leq 5$
- (⇐). Recall the trick.
  - $\sum_{j \in N} a_j x_j \leq b \Rightarrow \delta = 1 \iff \sum_{j \in N} a_j x_j - (m - \epsilon)\delta \geq b + \epsilon$
  - $2x_1 + 3x_2 + 1.1\delta \geq 1.1$

\[
2x_1 + 3x_2 + 4\delta \leq 5 \\
2x_1 + 3x_2 + 1.1\delta \geq 1.1
\]
A More Realistic Example

- **PPP**—Production Planning Problem. (A simple linear program).
- An engineering plant can produce five types of products: $p_1, p_2, \ldots, p_5$ by using two production processes: grinding and drilling. Each product requires the following number of hours of each process, and contributes the following amount (in hundreds of dollars) to the net total profit.

<table>
<thead>
<tr>
<th></th>
<th>$p_1$</th>
<th>$p_2$</th>
<th>$p_3$</th>
<th>$p_4$</th>
<th>$p_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grinding</td>
<td>12</td>
<td>20</td>
<td>0</td>
<td>25</td>
<td>15</td>
</tr>
<tr>
<td>Drilling</td>
<td>10</td>
<td>8</td>
<td>16</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Profit</td>
<td>55</td>
<td>60</td>
<td>35</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

PPP – More Info

- Each unit of each product take 20 manhours for final assembly.
- The factory has three grinding machines and two drilling machines.
- The factory works a six day week with two shifts of 8 hours/day. Eight workers are employed in assembly, each working one shift per day.
PPP

maximize

\[ 55x_1 + 60x_2 + 35x_3 + 40x_4 + 20x_5 \] (Profit/week)

subject to

\[ 12x_1 + 20x_2 + 0x_3 + 25x_4 + 15x_5 \leq 288 \] (Grinding)
\[ 10x_1 + 8x_2 + 16x_3 + 0x_4 + 0x_5 \leq 192 \] (Drilling)
\[ 20x_1 + 20x_2 + 20x_3 + 20x_4 + 20x_5 \leq 384 \] Final Assembly

\[ x_i \geq 0 \quad \forall i = 1, 2, \ldots 5 \]

Another PPP Modeling Example

- Let’s model the following situation.
  - If we manufacture $P_1$ or $P_2$ (or both), then at least one of $P_3$, $P_4$, $P_5$ must also be manufactured.

- We first need an indicator variable $z_j$ that indicate when each of the $x_j > 0$.
  - How do we model $x_j > 0 \Rightarrow z_j = 1$?
  - **Hint:** This is equivalent to $z_j = 0 \Rightarrow x_j = 0$
Modeling the Logic

Answer: \( x_j \leq Mz_j \)

- Given that we have included the constraints \( x_j \leq Mz_j \), we’d like to model the following implication:
  - \( z_1 + z_2 \geq 1 \Rightarrow z_3 + z_4 + z_5 \geq 1 \)
- Can you just “see” the answer?
- I can’t. So let’s try our “formulaic” approach.
- Think of it in two steps
  - \( z_1 + z_2 \geq 1 \Rightarrow \delta = 1 \)
  - \( \delta = 1 \Rightarrow z_3 + z_4 + z_5 \geq 1 \).

Look up the Tricks

- First we model the following:
  - \( z_1 + z_2 \geq 1 \Rightarrow \delta = 1 \)
- The formula from the bag o’ tricks
  - \( \sum_{j \in N} a_jx_j \geq b \Rightarrow \delta = 1 \iff \sum_{j \in N} a_jx_j - (M + \epsilon)\delta \leq b - \epsilon \)
- \( M \): Upper Bound on \( \sum_{j \in N} a_jz_j - b \)
  - \( M = 1 \) in this case. (\( z_1 \leq 1, z_2 \leq 1, b = 1 \)).
- \( \epsilon \): “Tolerance Level” indicating the minimum about by which the constraint can be violated.
  - \( \epsilon = 1 \) in this case!
  - If the constraint is going to be violated, then it will be violated by at least one.
Modeling $z_1 + z_2 \geq 1 \Rightarrow \delta = 1$, Cont.

- Just plug in the formula $\sum_{j \in N} a_j x_j - (M + \epsilon)\delta \leq b - \epsilon$
  - $z_1 + z_2 - 2\delta \leq 0$
  - Does this do what we want?

<table>
<thead>
<tr>
<th>$z_1$</th>
<th>$z_2$</th>
<th>$\delta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$\geq 0$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$\geq 1/2$ ($\Rightarrow = 1$)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$\geq 1/2$ ($\Rightarrow = 1$)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$\geq 1$</td>
</tr>
</tbody>
</table>

Second Part

- Want to model the following:
  - $\delta = 1 \Rightarrow z_3 + z_4 + z_5 \geq 1$.

- The formula from the bag o’ tricks
  - $\delta = 1 \Rightarrow \sum_{j \in N} a_j x_j \geq b \iff \sum_{j \in N} a_j x_j + m\delta \geq m + b$

- $m$ : lower bound on $\sum_{j \in N} a_j x_j - b$.
  - $m = -1$. ($z_1 \geq 0, z_2 \geq 0, b = 1$).

- Plug in the formula:
  - $z_3 + z_4 + z_5 - \delta \geq 0$

- It works! (Check for $\delta = 0, \delta = 1$).
PPP, Make 1 or 2 ⇒ make 3, 4, or 5

maximize

\[ 55x_1 + 60x_2 + 35x_3 + 40x_4 + 20x_5 \quad \text{(Profit/week)} \]

subject to

\[ \begin{align*}
12x_1 + 20x_2 + 0x_3 + 25x_4 + 15x_5 & \leq 288 \\
10x_1 + 8x_2 + 16x_3 + 0x_4 + 0x_5 & \leq 192 \\
20x_1 + 20x_2 + 20x_3 + 20x_4 + 20x_5 & \leq 384 \\
\end{align*} \]

\[ x_i \leq M_i z_i \quad \forall i = 1, 2, \ldots 5 \]

\[ z_1 + z_2 - 2\delta \leq 0 \]

\[ z_3 + z_4 + z_5 - \delta \geq 0 \]

\[ x_i \geq 0 \quad \forall i = 1, 2, \ldots 5 \]

\[ z_i \in \{0, 1\} \forall i = 1, 2, \ldots 5 \]

\[ \delta \in \{0, 1\} \]

---

Cool Things You Can Now Do

- Either constraint 1 or constraint 2 must hold
  - Create indicators \( \delta_1, \delta_2 \), then \( \delta_1 + \delta_2 \geq 1 \)
- At least one constraint of all the constraints in \( M \) should hold
  - \( \sum_{i \in M} \delta_i \geq 1 \)
- At least \( k \) of the constraints in \( M \) must hold
  - \( \sum_{i \in M} \delta_i \geq k \)
- If \( x \), then \( y \)
  - \( \delta_y \geq \delta_x \)
That’s All Folks!

- That’s it for modeling.
- You should have read N&W I.1.1 by now.
- Those of you who want to get a jump start on the homeworks can consider doing problems
  - ???
  - ???

That’s it for modeling. You should have read N&W I.1.1 by now. Those of you who want to get a jump start on the homeworks can consider doing problems.

- ???
- ???