Lecture 8 Constrained optimization and integer programming

Weinan E$^{1,2}$ and Tiejun Li$^2$

$^1$Department of Mathematics,
Princeton University,
weinan@princeton.edu

$^2$School of Mathematical Sciences,
Peking University,
tieli@pku.edu.cn
No.1 Science Building, 1575
Outline

Examples

Constrained optimization

Integer programming
Suppose an investor own a block of $S$ shares that we want to sell over the next $N$ days. The total expected value of our shares is

$$V(s) = \sum_{t=1}^{N} p_t s_t$$

where $(s_1, \cdots, s_N)$ is the amount that we sell on each day and $(p_1, \cdots, p_N)$ are the prices on each day. Moreover, the price $p_t$ follows the following dynamics

$$p_t = p_{t-1} + \alpha s_t, \quad t = 1, \cdots, N$$

How should the investor sell his block of shares?

Mathematical formulation:

$$\max \sum_{t=1}^{N} p_t s_t$$

Subject to the constraint

$$\sum_{t=1}^{N} s_t = S, \quad p_t = p_{t-1} + \alpha s_t, \quad s_t \geq 0, \quad t = 1, \cdots, N;$$

A constrained nonlinear optimization.
0-1 Knapsack problem

- The thief wants to steal $n$ items. The $i$-th item weights $w_i$ and has value $v_i$. The problem is to take most valuable load with limit of weight $W$.

- Mathematical formulation:

$$\max V = \sum_{j=1}^{n} v_j x_j$$

$$\sum_{j=1}^{n} w_j x_j \leq W$$

$$x_j = 0 \text{ or } 1, \quad j = 1, \ldots, n$$

- $x_j$ must be integers. An integer programming problem.
Assignment problem

- Assign $n$ persons to finish $n$ jobs. The cost for the $i$-th person to do $j$-th job is $c_{ij}$. Find the optimal assignment procedure to minimize the cost.
- Mathematical formulation: Define $x_{ij} = 1$ if the $i$-th person does $j$-th job, and $x_{ij} = 0$ otherwise, then

$$\max z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

$$\sum_{j=1}^{n} x_{ij} = 1, i = 1, \ldots, n$$

$$\sum_{i=1}^{n} x_{ij} = 1, j = 1, \ldots, n$$

$x_{ij} = 0$ or $1$, $i, j = 1, \ldots, n$

- A 0-1 integer programming problem.
Outline

Examples

Constrained optimization

Integer programming
General formulation for constrained nonlinear optimization

▶ General form

\[ \begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g_i(x) \leq 0, \quad i = 1, 2, \ldots, m \\
& \quad h_j(x) = 0, \quad j = 1, 2, \ldots, p \\
& \quad x \in X \subset \mathbb{R}^n, \quad x = (x_1, x_2, \ldots, x_n)
\end{align*} \]

and call the set

\[ S = \left\{ x \mid g_i(x) \leq 0, \ i = 1, 2, \ldots, m; \ h_j(x) = 0, \ j = 1, 2, \ldots, p; \ x \in X \right\} \]

the feasible solution of the problem.
Penalty method

- The idea of penalty method is to convert the constrained optimization problem into an unconstrained optimization problem by introducing a penalty term.
- Define the penalty function

\[ F(x, M) = f(x) + Mp(x) \]

\( M > 0 \) is called penalty factor, \( p(x) \) is called penalty term. In general \( p(x) \geq 0 \) for arbitrary \( x \in \mathbb{R}^n \) and \( p(x) = 0 \) iff \( x \in S \).
Penalty method

- For equality constrain define
  \[ g_j^+(x) = (h_j(x))^2, \quad j = 1, 2, \ldots, p \]

  and for inequality constrain define
  \[ g_{i+p}^+(x) = \begin{cases} 
  0, & g_i(x) \leq 0 \\
  (g_i(x))^2, & g_i(x) > 0 
  \end{cases} \]

  for \( i=1,2,\ldots,m \).

- Define \( L = p + m \) and the penalty function
  \[ F(x, M_k) = f(x) + M_k \sum_{i=1}^{L} g_i^+(x) \]

  where \( M_k > 0 \) and
  \[ M_1 < M_2 < \cdots < M_k < \cdots \rightarrow +\infty \]
Penalty method

- If $M_k \gg 1$ and if the penalty function
  
  $$F(x, M_k) = f(x) + M_k p(x)$$

  is minimized, this will force the penalty term
  
  $$p(x) \approx 0.$$ 

  Otherwise $M_k$ will amplify many times!! That’s why it is called penalty method.

- In general, take
  
  $$M_{k+1} = cM_k, \quad c \in [4, 50]$$
Algorithm for penalty method

1. Take $M_1 > 0$, tolerance $\epsilon > 0$, Initial state $x_0$, set $k = 1$;

2. Solve the unconstrained optimization

$$
\min F(x, M_k) = f(x) + M_k \sum_{i=1}^{L} g_i^+(x)
$$

with initial data $x_{k-1}$, and the solution is $x_k$;

3. Define

$$
\tau_1 = \max\{|h_i(x_k)|\}, \quad \tau_2 = \max\{g_i(x_k)\}
$$

and $\tau = \max\{\tau_1, \tau_2\}$;

4. If $\tau < \epsilon$, over; otherwise, set $M_{k+1} = cM_k$, $k = k + 1$, return to step 2.
Barrier method

- Barrier method is suitable for optimization as

\[
\min f(x), \quad \text{s.t. } x \in S
\]

where \( S \) is a set characterized only by inequality constraints

\[
S = \left\{ x | g_i(x) \leq 0, \quad i = 1, 2, \ldots, m \right\}
\]

- Graphical interpretation of barrier method
Barrier method

- Define barrier term $B(x)$ such that

\[ B(x) \geq 0 \quad \text{and} \quad B(x) \to \infty \quad \text{as} \quad x \to \text{boundary of } S \]

- Inverse barrier term

\[ B(x) = \sum_{i=1}^{m} g_i^+(x) \]

and

\[ g_i^+(x) = -\frac{1}{g_i(x)} \]

- Logarithmic barrier term

\[ g_i^+(x) = -\ln(-g_i(x)) \]

- Barrier function

\[ F(x, r_k) = f(x) + r_k B(x) \]

where

\[ r_k > 0, \quad r_1 > r_2 > \cdots > r_k > \cdots \to 0. \]
Barrier method

- Though the formulation of barrier method

\[ F(x, r) = f(x) + rB(x), x \in S \]

is still a constrained optimization, but the property

\[ F(x, r) \to \infty \text{ as } x \to \text{ boundary of } S \]

makes the numerical implementation an unconstrained problem.

- The implementation will be an iteration \((c \in [4, 10])\)

\[ r_{k+1} = r_k / c \]

until some type of convergence criterion is satisfied.
Outline

Examples

Constrained optimization

Integer programming
Discrete optimization

- Integer programming is a typical case in discrete optimization. There are large amount of discrete optimization problems in graph theory and computer science.
- Discrete optimization models are, except for some special cases, are extremely hard to solve in practice. They are NP-Hard problem. (Is \( NP=\overline{P} \)? This is a million dollar problem.)
- Unfortunately there are no general widely applicable methods for solving discrete problems. But there are some common themes such as relaxation, branch-and-bound etc.
- There are some heuristic ideas such as local search methods, simulated annealing, genetic algorithms etc.
Integer linear programming

- General form

  \[
  \max z = c^T x \\
  Ax \leq b, \quad x \geq 0, \quad x_i \in I, \quad i \in J \subset \{1, 2, \cdots, n\}
  \]

  where

  \[
  x = (x_1, x_2, \cdots, x_n), \quad c = (c_1, c_2, \cdots, c_n) \\
  b = (b_1, b_2, \cdots, b_m), \quad A = (a_{ij})_{m \times n}, \quad I = \{0, 1, 2, \cdots\}
  \]

  If \( J = \{1, 2, \cdots, n\} \), it is a pure integer programming. If \( J \neq \{1, 2, \cdots, n\} \), it is a mixed integer programming problem.
Relaxation and decomposition

- **Relaxation**: the problem obtained after relaxing some constrained condition is called relaxation problem of the primitive problem. For example we obtain the linear programming after relaxing the integer constraints.

- **Decomposition**: define $R(P)$ the feasible solution set of problem $(P)$. If

\[
\bigcup_{i=1}^{m} R(P_i) = R(P)
\]

\[
R(P_i) \cap R(P_j) = \emptyset \quad (1 \leq i \neq j \leq m),
\]

we call the subproblems $(P_1), (P_2), \cdots, (P_m)$ a decomposition of $(P)$. 
Example

Example

\[ \text{max } z = 5x_1 + 8x_2 \]
\[ x_1 + x_2 \leq 6, \quad 5x_1 + 9x_2 \leq 45 \]
\[ x_1, x_2 \in I = \mathbb{N} \cup \{0\} \]

Relaxation: let \( x_1, x_2 \geq 0 \), it is a linear programming problem, the optimum is \( x = (2.25, 3.75) \) which does NOT belong to \( I \)!

Decomposition: decompose the range of \( x_2 \) into

\[ x_2 \geq 4 \quad \text{or} \quad x_2 \leq 3. \]

We obtain two subproblems.
The basic framework of Branch-and-bound method is as follows

1. **Upper Bounds**: Efficient methods for determining a good upper bound $UB(P)$;

2. **Branching Rules**: Methods for replacing an instance $(P)$ of the discrete optimization problem with some further “smaller” subproblems $(P_l)$ such that some optimal solution of $(P)$ maps to an optimal solution of a subproblem $(P_l)$.

3. **Lower Bounds**: Efficient heuristics that attempt to determine a feasible candidate solution $S$ with as low a value as is practical, yielding the lower bound $LB(P)$. 
Some definitions

- Define the floor and ceiling function for any \( a \in \mathbb{R} \)

  \[
  \lfloor a \rfloor := \text{The integer nearest to } a \text{ but less than } a
  \]

  \[
  \lceil a \rceil := \text{The integer nearest to } a \text{ but bigger than } a
  \]

  It's clear that

  \[
  0 \leq a - \lfloor a \rfloor < 1, \quad 0 \leq \lceil a \rceil - a < 1
  \]

- Examples

  \[
  \lfloor -\frac{1}{7} \rfloor = -1, \quad \lceil \frac{1}{28} \rceil = 0, \quad \lceil \frac{7}{4} \rceil = 1
  \]

  \[
  \lfloor -\frac{1}{7} \rfloor = 0, \quad \lceil \frac{1}{28} \rceil = 1, \quad \lceil \frac{7}{4} \rceil = 2
  \]
Branching rules

- Define the optimal solution of the linear-programming relaxation as

\[ \mathbf{x}^* = (x_1, x_2, \ldots, x_n) \]

- Branching rule: We choose a variable \( x_k^* \notin \mathbb{Z} \). We branch by creating two new subproblems:

1. \((P')\) together with the additional inequality

\[ x_k \leq \lfloor x_k^* \rfloor \]

2. \((P')\) together with the additional inequality

\[ x_k \geq \lceil x_k^* \rceil \]
Branch-and-bound: an example

▶ Example

\[ \text{max } z = -x_1 + x_2 \]

Subject to

\[ 12x_1 + 11x_2 \leq 63 \]
\[ -22x_1 + 4x_2 \leq -33 \]
\[ x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{Z} \]

\[ z=1.29 \]
Branch-and-bound: an example

- First solve the relaxation problem we have

<table>
<thead>
<tr>
<th>Subprogram</th>
<th>$z^*$</th>
<th>$x_1^*$</th>
<th>$x_2^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP</td>
<td>1.29</td>
<td>2.12</td>
<td>3.41</td>
</tr>
</tbody>
</table>

Then the lower and upper bounds

$$LB = -\infty, \quad UB = 1.29$$

- Branching $x_1$ we have two subprograms and solve the relaxation problems respectively

<table>
<thead>
<tr>
<th>Subprogram</th>
<th>$z^*$</th>
<th>$x_1^*$</th>
<th>$x_2^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP with $x_1 \leq 2$</td>
<td>0.75</td>
<td>2.00</td>
<td>2.75</td>
</tr>
<tr>
<td>IP with $x_1 \geq 3$</td>
<td>-0.55</td>
<td>3.00</td>
<td>2.45</td>
</tr>
</tbody>
</table>
Branch-and-bound: an example

\[ z = -0.55 \]
\[ z = 0.75 \]
Branch-and-bound: an example

- Branching $x_2$ of IP with $x_1 \leq 2$ and solve the relaxation problem

<table>
<thead>
<tr>
<th>Subprogram</th>
<th>$z^*$</th>
<th>$x_1^*$</th>
<th>$x_2^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP with $x_1 \leq 2, x_2 \leq 2$</td>
<td>0.14</td>
<td>1.86</td>
<td>2.00</td>
</tr>
<tr>
<td>IP with $x_1 \leq 2, x_2 \geq 3$</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- Thus we have subprograms

<table>
<thead>
<tr>
<th>Subprogram</th>
<th>$z^*$</th>
<th>$x_1^*$</th>
<th>$x_2^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP with $x_1 \geq 3$</td>
<td>-0.55</td>
<td>3.00</td>
<td>2.45</td>
</tr>
<tr>
<td>IP with $x_1 \leq 2, x_2 \leq 2$</td>
<td>0.14</td>
<td>1.86</td>
<td>2.00</td>
</tr>
</tbody>
</table>

and

$LB = -\infty$, $UB = 0.14$
Branch-and-bound: an example

- Branching $x_1$ of IP with $x_1 \leq 2, x_2 \leq 2$ we have

<table>
<thead>
<tr>
<th>Subprogram</th>
<th>$z^*$</th>
<th>$x_1^*$</th>
<th>$x_2^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP with $x_1 \leq 2, x_2 \leq 2, x_1 \leq 1$</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>IP with $x_1 \leq 2, x_2 \leq 2, x_1 \geq 2$</td>
<td>0.00</td>
<td>2.00</td>
<td>2.00</td>
</tr>
</tbody>
</table>

and because $x^* \in \mathbb{Z}$ in the subprogram, we have

\[ LB = 0.00, \quad UB = 0.14 \]

- Because $-0.55 < LB = 0.00$, the subprogram

\[ \text{IP with } x_1 \geq 3 \]

is deleted.

- So finally we have the optimal solution

\[ x^* = (2, 2), \quad z^* = 0.00 \]
A cutting-plane is a **linear inequality that is generated as needed** in the course of solving an integer linear program as a sequence of linear programs.

**Generic cutting-plane method**

1. Initially let $LP$ be the linear programming relaxation of $IP$;
2. Let $x^*$ be an optimal extreme-point solution of $LP$;
3. If $x^*$ is all integer, then stop because $x^*$ is optimal to $IP$;
4. If $x^*$ is not all integer, then find an inequality that is satisfied by all feasible solutions of $IP$, but is violated by $x^*$, append the inequality to $LP$, and go to step 2.
Gomory cutting-plane

- For an equality constraints

\[ x_1 + (-\frac{1}{7})x_3 + \frac{1}{28} x_4 = \frac{7}{4} \]

Perform transformation

\[
\lfloor 1 \rfloor x_1 + \lfloor -\frac{1}{7} \rfloor x_3 + \lfloor \frac{1}{28} \rfloor x_4 - \lfloor \frac{7}{4} \rfloor = (\lfloor 1 \rfloor - 1)x_1
\]

\[
+ (\lfloor -\frac{1}{7} \rfloor + \frac{1}{7})x_3 + (\lfloor \frac{1}{28} \rfloor - \frac{1}{28})x_4 + \frac{7}{4} - \lfloor \frac{7}{4} \rfloor
\]
Gomory cutting-plane

- We have
  \[ x_1 - x_3 - 1 = -\frac{6}{7} x_3 - \frac{1}{28} x_4 + \frac{3}{4} \]

- Because
  1. \( x_1, x_3 \) are integers from the left hand side;
  2. \( x_3, x_4 \in \mathbb{N} \cup \{0\} \) from the righthand side;

we have the cutting plane

\[ -\frac{6}{7} x_3 - \frac{1}{28} x_4 + \frac{3}{4} \leq 0 \]

or equivalently

\[ x_1 - x_3 - 1 \leq 0 \]

- Generating the inequality from the lower floor decomposition technique is called Gomory cutting plane method.
Concrete example of Gomory cutting plane method

Example

\[ \begin{align*}
\min z &= -x_1 - 27x_2 \\
-x_1 + x_2 &\leq 1 \\
24x_1 + 4x_2 &\leq 25 \\
x_1, x_2 &\geq 0, x_1, x_2 \in I
\end{align*} \]

Transform into standard form and make relaxation

\[ \begin{align*}
\min z &= -x_1 - 27x_2 \\
-x_1 + x_2 + x_3 &= 1 \\
24x_1 + 4x_2 + x_4 &= 25 \\
x_1, x_2 &\geq 0
\end{align*} \]
Concrete example of Gomory cutting plane method

- Simplex method for optimal solution

<table>
<thead>
<tr>
<th>Basis</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_2$</td>
<td>0</td>
<td>1</td>
<td>$\frac{6}{7}$</td>
<td>$\frac{1}{28}$</td>
<td>$\frac{7}{4}$</td>
</tr>
<tr>
<td>$a_1$</td>
<td>1</td>
<td>0</td>
<td>$-\frac{1}{7}$</td>
<td>$\frac{1}{28}$</td>
<td>$\frac{3}{4}$</td>
</tr>
</tbody>
</table>

i.e. we have

$$x^* = \left( \frac{3}{4}, \frac{7}{4} \right)$$

- $x^* \notin \mathbb{Z}$, we determine the cutting plane

$$\frac{3}{4} - \frac{6}{7}x_3 - \frac{1}{28}x_4 \leq 0$$

Transform into standard form we have

$$-24x_3 - x_4 + x_5 = -21$$

And supplement this constraint into the primitive constraints.
Concrete example of Gomory cutting plane method

- Simplex method for optimal solution

<table>
<thead>
<tr>
<th>Basis</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>$a_5$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>$\frac{1}{28}$</td>
<td>1</td>
</tr>
<tr>
<td>$a_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>$\frac{1}{24}$</td>
<td>$\frac{1}{168}$</td>
<td>$\frac{7}{8}$</td>
</tr>
<tr>
<td>$a_3$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>$\frac{1}{24}$</td>
<td>$\frac{1}{24}$</td>
<td>$\frac{7}{8}$</td>
</tr>
</tbody>
</table>

i.e. we have

$$x^* = \left(\frac{7}{8}, 1\right)$$

- $x^* \notin \mathbb{Z}$, we determine the cutting plane

$$\frac{7}{8} - \frac{1}{24}x_4 - \frac{23}{24}x_5 \leq 0$$

Transform into standard form we have

$$-x_4 - 23x_5 + x_6 = -21$$

And supplement this constraint into the primitive constraints.
Concrete example of Gomory cutting plane method

- Simplex method for optimal solution

<table>
<thead>
<tr>
<th>Basis</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>$a_5$</th>
<th>$a_6$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>$\frac{1}{28}$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$a_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$-\frac{27}{28}$</td>
<td>$\frac{1}{24}$</td>
<td>0</td>
</tr>
<tr>
<td>$a_3$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$-1$</td>
<td>$\frac{1}{24}$</td>
<td>0</td>
</tr>
<tr>
<td>$a_4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>23</td>
<td>$-1$</td>
<td>21</td>
</tr>
</tbody>
</table>

i.e. we have

$$x^* = (0, 1)$$

- $x^* \in \mathbb{Z}$, so we obtain the optimal solution

$$x^* = (0, 1), \quad z^* = -27$$
The geometric meaning of Gomory cutting plane

Transforming the cutting plane into planes with primitive variables $x_1, x_2$, we have the cutting plane equations

**Cutting plane 1** $x_2 \leq 1$

**Cutting plane 2** $x_1 + 27x_2 \leq 27$