Section #2: Linear and Integer Programming

Prof. Dr. Sven Seuken
8.3.2012

(with most slides borrowed from David Parkes)
Housekeeping

- Game Theory homework submitted?
- HW-00 and HW-01 returned
- Feedback on Comprehension Questions later today or tomorrow
- BitTorrent assignment → online this afternoon
- Next chapter → online this afternoon
- Questions? Concerns?

Prof. Dr. Sven Seuken - University of Zurich
Outline

1. $O()$-Notation
2. NP vs. P
3. Linear Programming
4. Integer Programming
Big O()-Notation

- Analysis of algorithms
- Asymptotic running time
- Input Size...?

\[ x = \# \text{ of bytes} \]

\[ O(x^2) \]

\[ m^2 \cdot 2.8 \]

\[ O(xm^2) \]

\[ O(m^4) \rightarrow O(xm^2) \]
Polynomial vs. Exponential

\[ O(x^2) \]

\[ O(x^3) \]

\[ O(x^n) = \text{polynomial} \]

\[ O(2^x), O(3^x) \]

\[ 100,000 - x^k + 1,000,000 \]

Prof. Dr. Sven Seuken - University of Zurich
The Complexity Class “P”

- Decision Problems
- A problem is in P, if there exists a worst-case polynomial time algorithm for solving the problem.
- Efficient

\[ O(x^k) \]
The Complexity Class “NP”

- Decision Problems! (YES vs NO)
- A problem is in NP, if, given an answer (or a proof), there exists a worst-case polynomial time algorithm that can verify that the answer is YES.
NP-hard vs. NP-complete

Problem \in \text{NP} \checkmark

Problem \textbf{NP-hard}.

Reduction: \text{P}_1 \leq_p \text{P}_2

NP-complete: \text{P}_1 \in \text{NP} \land \text{P}_1 \text{ is \text{NP-hard}}

\forall \text{P}_i, \text{P}_j \in \text{NP-c}: \text{P}_i \leq_p \text{P}_j

Prof. Dr. Sven Seuken - University of Zurich
P vs. NP

• Important Conjecture: P ≠ NP

• Why do we believe this?

• Consequence:
  – Fastest algorithms for problems that are NP-hard probably require exponential time in the worst case.

→ Inefficient algorithms (for medium-sized problems)
Mathematical Optimization

• What is “optimization”?  
  – problem of making decisions to maximize or minimize an objective in the presence of complicating constraints

• Examples: scheduling aircraft, designing a jet engine, sourcing goods, predicting who will win a football game.
Example: Going Shopping

• I can buy pizza $x_1$, and chicken $x_2$
• Pizza costs $5 per unit, and gives me 3 carbohydrates and 2 proteins per unit
• Chicken costs $10 per unit, and gives me 5 proteins and 1 carbohydrate per unit
• Task:
  – Maximize the carbohydrates
  – Do not spend more than $30
  – Need at least 10 protein units
• Variables?
• Objective?
• Constraints?

\[
\begin{align*}
\text{max} & \quad 3x_1 + x_2 \\
\text{s.t.} & \quad 5x_1 + 10x_2 \leq 30 \\
& \quad 2x_1 + 5x_2 \geq 10 \\
& \quad x_1, x_2 \geq 0
\end{align*}
\]
Example: Marketing campaign

• Comedy – 7 million high-income women, 2 million high-income men. Cost $50,000
• Football – 2 million high-income women and 12 million high-income men. Cost $100,000
• Goal: reach at least 28 million high-income women and 24 million high-income men at MINIMAL cost

\[
\begin{align*}
\text{max} & \quad x_1 \cdot x_2 \\
\text{max} & \quad x_1^2 \\
\min & \quad 50,000x_1 + 100,000x_2 \\
\text{s.t.} & \quad 7x_1 + 2x_2 \geq 28 \\
& \quad 2x_1 + 12x_2 \geq 34 \\
& \quad x_1, x_2 \geq 0
\end{align*}
\]
graphical version of problem
(solution is $x_1=3.6$, $x_2=1.4$, value 320)

\[ \begin{align*}
\min \quad & z = 50x_1 + 100x_2 \\
\text{s.t.} \quad & 7x_1 + 2x_2 \geq 28 \\
& 2x_1 + 12x_2 \geq 24 \\
& x_1, x_2 \geq 0
\end{align*} \]
Example: Unbounded objective

\begin{align*}
\text{max} \quad & z = -x_1 + x_2 \\
\text{s.t.} \quad & -x_1 + 4x_2 \geq 0 \\
\quad & x_1 \leq 4 \\
\quad & x_1, x_2 \geq 0
\end{align*}
Example: Infeasible problem

\[
\begin{align*}
\text{max} \quad & z = x_1 + x_2 \\
\text{s.t.} \quad & 3x_1 + x_2 \geq 6 \\
& 3x_1 + x_2 \leq 3 \\
& x_1, x_2 \geq 0
\end{align*}
\]
Union rules state that each full-time employee must work 5 consecutive days and then receive 2 days off.

Formulate an LP to minimize the number of full-time employees who must be hired.
\[
\begin{align*}
\min z &= \sum_i x_i \\
\text{s.t.} \quad & \sum_i x_i - x_2 - x_3 \geq 17 \\
& \sum_i x_i - x_3 - x_4 \geq 13 \\
& \sum_i x_i - x_4 - x_5 \geq 15 \\
& \sum_i x_i - x_5 - x_6 \geq 19 \\
& \sum_i x_i - x_6 - x_7 \geq 14 \\
& \sum_i x_i - x_7 - x_1 \geq 16 \\
& \sum_i x_i - x_1 - x_2 \geq 11 \\
x_i & \geq 0
\end{align*}
\]

Note: will need solution to be integral!
\[ \text{max/min} \quad x, \ldots \]

\[ \text{s.t.} \]

\[ \text{max} \quad x_1 + x_2 \]

\[ \text{s.t.} \quad x_1 \leq 3 \quad x_1, x_2 \geq 0 \quad \rightarrow \text{Linear Program} \]

\[ x_2 \leq 5 \quad x_1, x_2 \leq 2 \quad \rightarrow \text{Integer Program} \]

\[ x_1, x_2 \geq 0 \]
Polyhedron

• **Definition.** A polyhedron is a set that can be described in form $P=\{x \in \mathbb{R}^n \mid Ax \geq b\}$

• **Fact:** feasible set of any LP can be described in this way, and in particular $\{x \in \mathbb{R}^n \mid Ax=b, x \geq 0\}$ is also a polyhedron.

• **Theorem.** Every polyhedron is a convex set.

• Why?
Extreme points
Algorithms for LPs

- Idea: Check every extreme point
  - With n constraints: $2^n$ extreme points
  - $\rightarrow$ takes exponential time

- Simplex Algorithm:
  - Move from extreme point to extreme point
  - Stop when local optimum (= global optimum)
  - Very fast in practice (e.g., more than 10,000 constraints)
  - Worst-case time complexity: exponential
Complexity of LPs?

• Solving an LP is in P!
• Thus, there exists a worst-case polynomial time algorithm for solving LPs: → interior-point method
• In the worst case faster than simplex method
• In the average case, slower than simplex...
• In practice:
  – CPLEX (IBM)
Integer Programming
Some uses of IP

• Scheduling problems
  – e.g., air taxi scheduling (assign each air taxi to a particular route)

• Strategic procurement
  – e.g., hospital system determining which suppliers to use for sourcing of medical/surgical equipment

• Electricity generation planning
  – e.g., developing a schedule to determine when to start-up plants, and levels to run at

• Clearing kidney exchanges
  – e.g., finding “donor chains”

• Computational biology
  – e.g. determining evolutionary trees (“phylogenetic trees”) from population variation data
Shouldn’t IPs be easier to solve

• “There are fewer feasible solutions”

---

What is misleading about this argument?
max $5x_1 + 8x_2$
\[ \text{s.t. } x_1 + x_2 \leq 6 \]
\[ 5x_1 + 9x_2 \leq 45 \]
\[ x_1, x_2 \geq 0, \text{ integer} \]

<table>
<thead>
<tr>
<th></th>
<th>cont</th>
<th>round off</th>
<th>nearest feas</th>
<th>integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>2.25</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$x_2$</td>
<td>3.75</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>$z$</td>
<td>41.25</td>
<td>infeas</td>
<td>34</td>
<td>40</td>
</tr>
</tbody>
</table>

optimal integer solution
$x^*=(0,5), z^*=40$

optimal continuous solution
$x^*=(9/4, 15/4), z^*=41.25$
Example: Weighted Knapsack

- E.g., Budget $b$ available for investment in projects. Project $j$ in $\{1,\ldots,n\}$ has cost ("size") $a_j$ and value ("weight") $c_j$.
- Let $x_j = 1$ if project $j$ selected, $x_j = 0$ otherwise.
- $\max \sum_j c_j x_j$ (maximize value)
- $\sum_j a_j x_j \leq b$ (budget constraint)
- $x_j \in \{0,1\}$ (no fractional projects)
Example: Traveling Salesman Problem (TSP)

- Salesman must visit each of $N=\{1,\ldots,n\}$ cities once and then return to starting point. Cost $c_{ij}\geq 0$ to travel from $i$ to $j$. Goal: find tour that minimizes total cost.

A feasible tour in a seven-city TSP

Examples: FedEx pick-up, machine placing modules on a circuit board, visiting colleges,...
• How many solutions?
• Starting at city 1, there are n-1 choices
• For the next city, n-2 choices,...
• (n-1)! feasible tours

<table>
<thead>
<tr>
<th>n</th>
<th>log n</th>
<th>n^{0.5}</th>
<th>n^2</th>
<th>2^n</th>
<th>n!</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>3.32</td>
<td>3.16</td>
<td>10^2</td>
<td>1.02x10^3</td>
<td>3.6x10^6</td>
</tr>
<tr>
<td>100</td>
<td>6.64</td>
<td>10.00</td>
<td>10^4</td>
<td>1.27x10^{30}</td>
<td>9.33x10^{157}</td>
</tr>
<tr>
<td>1000</td>
<td>9.97</td>
<td>31.62</td>
<td>10^6</td>
<td>1.07x10^{301}</td>
<td>4.02x10^{2567}</td>
</tr>
</tbody>
</table>
Branch and Bound: Solving IPs

• Ignore integrality constraints, solve the LP relaxation and hope the solution is integer!

• Example:
  \[
  \begin{align*}
  \text{max} & \quad 5x_1 + 8x_2 \\
  \text{s.t.} & \quad x_1 + x_2 \leq 6 \\
  & \quad 5x_1 + 9x_2 \leq 45 \\
  & \quad x_1, \quad x_2 \geq 0, \text{ integer}
  \end{align*}
  \]

• Let \( S_0 \) denote feasible solution space of IP

• Obtain the initial LP relaxation by dropping the integrality constraints
\[ x_1 + x_2 \leq 6 \]

\[ 5x_1 + 9x_2 \leq 45 \]

LP optimal
\[ x_0 = (2.25, 3.75) \]
\[ z_0 = 41.25 \]
Let $P_0 = \text{LP}(S_0) = \{x \in \mathbb{R}^2: x_1 + x_2 \leq 6, 5x_1 + 9x_2 \leq 45, x \geq 0\}. \ z_0 = 41.25, x_0 = (2.25, 3.75)$

Consider variable $x_2$. Will be integer in solution to IP. Add constraints $x_2 \leq 3, x_2 \geq 4$.

Let $S_1$ denote $S_0 \bigcap \{x_2 \leq 3\}; \ P_1 = \text{LP}(S_1)$

Let $S_2$ denote $S_0 \bigcap \{x_2 \geq 4\}; \ P_2 = \text{LP}(S_2)$

Can plot the feasible region for $P_1$ and $P_2$
\[ x_1 + x_2 \geq 6 \]
\[ 5x_1 + 9x_2 \geq 45 \]
• Study two new IPs, with feasible solution spaces $S_1$ and $S_2$.

• Call $S_1$ and $S_2$ subproblems and this process branching.

• **Consider** $S_1$ first. Optimal solution to LP($S_1$) is $x_1=(3,3)$, $z_1=39$. Especially helpful! Our first feasible solution; write $x^*_1=(3,3)$, $z^*_1=39$.

• Update the incumbent, $z:=\max(z, z^*_1) = \max(-1, 39) = 39$, and $x:=(3,3)$. 

---

Prof. Dr. Sven Seuken - University of Zurich
• Now consider $S_2$. Optimal solution to LP($S_2$) is fractional.
• $\overline{x}_2=(1.8,4)$, $z_2=41$
• If $\overline{z}_2 \cdot \overline{z} = 39$ would be done... any integer solution in $S_2$ would be smaller yet.
• But, not the case. Divide $S_2$ into two subproblems
  • $S_3 = S_0 \Delta \{x_2 >= 4, x_1 >= 1\}$
  • $S_4 = S_0 \Delta \{x_2 >= 4, x_1 >= 2\}$
• $S=S_1 [ S_3 [ S_4, S_1 \Delta S_3 \Delta S_4 ] ]$;
• Develop a **search tree** of IP subproblems

\[
\begin{align*}
S_0 & : x_2 \leq 3 \quad \text{solved } x^*_1 = (3,3) \\
S_1 & : x_2 \geq 4 \\
S_2 & : x_1 \leq 1 \quad x_1 \geq 2 \\
S_3 & : 39 \\
S_4 & : 41.25
\end{align*}
\]

• Consider left branch. Solve LP(S₃)
\[ x_1 + x_2 \leq 6 \]

\[ 5x_1 + 9x_2 \leq 45 \]
• Consider left branch. Solve LP(S₃)
• \( \bar{x}_3 = (1, 4 \ 4/9), \ z_3 = 40 \ 5/9 \)
• Fractional, and \( z_3 \rightarrow z = 39 \ldots \) again subdivide and continue
• \( S_5 = S_0 \ \Delta \ {x_2 \cdot 4, \ x_1 \cdot 1, \ x_2 \cdot 4} = S_0 \ \Delta \ {x_1 \cdot 1, \ x_2 = 4} \)
• \( S_6 = S_0 \ \Delta \ {x_2 \cdot 4, \ x_1 \cdot 1, \ x_2 \cdot 5} = S_0 \ \Delta \ {x_1 \cdot 1, \ x_2 \cdot 5} \)
Prof. Dr. Sven Seuken - University of Zurich
• At this point have 3 possible directions.
• Could consider subproblems $S_4$, $S_5$, or $S_6$
• To be specific in the example, consider depth first search and consider a child of the most recently considered subproblem; and take the left branch.
• Solve LP($S_5$).
• $\overline{x}_5 = (1,4)$, $\overline{z}_5 = 37$
• $\overline{z}_5 \cdot z \geq 39$. Ignore branch, any feasible solution is bounded above by 37.
• Say that this node is “pruned by bound.”
• (Hence “branch and bound”).
S\_0 \quad X_2 \leq 3 \quad X_2 \geq 4

S\_1 \quad X_2 \leq 3 \\
\text{solved } x^*_1 = (3,3)

S\_2 \quad X_1 \leq 1 \quad X_1 \geq 2

S\_3 \quad X_2 \leq 4 \\
40 \ 5/9

S\_4 \quad X_2 \geq 5

S\_5 \quad 37 \\
\text{pruned by bound}

S\_6 \quad 37

Prof. Dr. Sven Seuken - University of Zurich
• Now at a leaf. Back up, find the first node with an unsolved problem.
• Right branch under $S_3$ (i.e., $S_6$)
• $z_6=40, x_6=(0,5)$. Another feasible solution!
• Write $z^*_6=40, x^*_6=(0,5)$
• Since $z^*_6 > z$, $z := \max(z, z^*_6) = \max(39,40)=40$, $x:=(0,5)$. A new incumbent.

• Finally, solve the problem under $S_2$ (i.e. $S_4$), and including any subproblems thereof. Infeasible.
Final search tree. Completely fathomed!
• Now completely **fathomed**. No branches left to explore.

• Can conclude that $x^* = x = (0,5)$, $z^* = z = 40$ is the optimal solution.
Complexity of IPs

• Integer Programming Problems are **NP-hard**

• Thus, it is likely that even the best possible algorithms for solving IPs will need **exponential time** in the worst case!

• Depending on the amount of structure in the problem, powerful algorithms (CPLEX) can solve problems with hundreds of variables and constraints