AM 121: Intro to Optimization Models and Methods

Lecture 14: Branch and Bound (II)

Yiling Chen
SEAS

Lesson Plan

- Branch and Bound review
- Node selection
- Branching decision
- Formulation strength
- CPLEX and “Cut generation”
Review

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

- Maintain a list of open subproblems, an incumbent $x$ with value $z$, and an upper bound $\bar{z}$ (maximal over bounds from parents of open nodes)
- At each step:
  - Node selection decision: pick an open subproblem and solve the LP relaxation
  - Try to fathom node (by infeasibility, by bound, or by integrality), update incumbent if better integral solution
  - Branching decision: if can’t fathom, pick a fractional variable in the solution to LPR and branch
- Continue until all open subproblems are fathomed, or the “optimality gap” is acceptable.
Node Selection Decision

- **Depth first (solve a node \( k \) just generated)**
  - Find an integer feasible solution quickly (or better solutions since feasible solutions tend to be deeper in a search tree), allow open subproblems to be fathomed by bounds
  - Can obtain next LP solution via dual simplex pivots, since simply add or refine an inequality on a variable.

- **Best-bound first (solve node \( k \) with highest LPR \( z_k \)).**
  - Will never solve a subproblem with an upper-bound less than the value of the optimal solution to the IP
  - Try to improve upper bounds quickly and prove optimality of current incumbent

- In practice: initial DFS (“diving”) followed by a mixture of best-bound and DFS.

DFS will solve one of \( S^5 \) or \( S^6 \)
“Best-bound first” will solve \( S^4 \) (since \( 41 > 40 \frac{5}{9} \))
Weak Duals

• Consider

  Primal: \( \max \{ c(x) : x \in X \} \)
  Dual: \( \min \{ w(y) : y \in Y \} \)

• Form a **weak dual pair** if \( c(x) \leq w(y), \ \forall x \in X, \forall y \in Y \)

• **Proposition.** \( \max \{ c^T x : Ax \leq b, x \in \mathbb{Z}^n_{\geq 0} \} \) and \( \min \{ b^T y : A^T y \geq c, y \in \mathbb{R}^m_{\geq 0} \} \) form a weak dual pair.

• Any feasible solution to the dual of the current (primal) LPR provides a valid upper bound.

Example: Dual Simplex Pivots

• Optimal tableau for original problem is

  \[
  z = 1.25s_1 + 0.75s_2 = 41.25 \\
  x_1 = 2.25s_1 - 0.25s_2 = 2.25 \\
  x_2 - 1.25s_1 + 0.25s_2 = 3.75 \quad (1)
  \]

• Consider \( P^2=LP(S^2) \) where \( S^2=S^0 \cap \{ x_2 \geq 4 \} \)

• Introduce excess variable \( e_1 \), and write \( x_2 \geq 4 \) as

  \[ x_2 - e_1 = 4 \quad (2) \]

• Establish basis \( B=\{x_1,x_2,e_1\} \) by \( (2)' = (1) - (2) \)

  \[ -1.25s_1 + 0.25s_2 + e_1 = -0.25 \quad (2') \]
Example: Dual pivots in BnB

• Initial tableau for $P^2$ is dual feasible:

\[
\begin{align*}
    z + 1.25s_1 + 0.75s_2 &= 41.25 \\
    x_1 + 2.25s_1 - 0.25s_2 &= 2.25 \\
    x_2 - 1.25s_1 + 0.25s_2 &= 3.75 \\
    -1.25s_1 + 0.25s_2 + e_1 &= -0.25
\end{align*}
\]

• Dual Pivot ($e_1$ out, $s_1$ in). Get:

\[
\begin{align*}
    z + s_2 + e_1 &= 41 \\
    x_1 + 0.2s_2 + 1.8e_1 &= 1.8 \\
    x_2 - e_1 &= 4 \\
    s_1 - 0.2s_2 - 0.8e_1 &= 0.2
\end{align*}
\]

• $B=\{x_1, x_2, s_1\}$, $x_1^*=1.8$, $x_2^*=4$, $z^*=41$

• One pivot! In general will take more than one pivot, but can often find a new LP solution quickly.

Branching Decision

• Most-fractional variable.
  – try to make progress towards integer feasible quickly

• Pseudocost method
  – Estimate the cost of forcing the variable to become an integer

• User priorities (e.g., “big decisions” first)
  – user might say “the location of a facility is more consequential than the districts it serves”

• Strong branching
  – do a little bit of “look ahead” before committing to a branching decision
Strong Branching: Example

Which most quickly improves the bound?
Answer: $x_2$ because it minimizes the maximal bound after branching decision.

Strong Branching Method

Let $C$ denote set of integer vars in current subproblem with a fractional assignment

For each $j \in C$:
(a) solve subproblem with $x_j \leq \lfloor x_j \rfloor$ and $x_j \geq \lceil x_j \rceil$
(b) let $\bar{z}_j^p$ and $\bar{z}_j^u$ denote the value of LP solns

Branch on $j^* = \arg\min_{j \in C} \max[\bar{z}_j^p, \bar{z}_j^u]$

Note: can also solve subproblems approx. in strong branching; use a few dual simplex pivots.
Advanced Comments

• If the LP relaxation is unbounded then:
  – If the basic integer variables take on integer values then the IP is unbounded
  – Else, branch.

• Branch and Bound will terminate if all integer variables are bounded, but need not o.w.

\[
\begin{align*}
\text{min} & \quad 0 \\
1 & \leq 3x - 3y \leq 2 \\
x, y & \in \mathbb{Z}
\end{align*}
\]

Infeasible, but B&B keeps searching

Formulations
Recall: Firehouse Location

Formulation \((P_1)\):
• If \(m=2\) sites, and \(n=4\) districts:
  • \(x_{11} + x_{21} + x_{31} + x_{41} \leq 4y_1\)
  • \(x_{12} + x_{22} + x_{32} + x_{42} \leq 4y_2\)
• \(m\) constraints

Alternate formulation \((P_2)\):
• \(x_{11} \leq y_1, x_{21} \leq y_1, x_{31} \leq y_1, x_{41} \leq y_1\)
• \(x_{12} \leq y_2, x_{22} \leq y_2, x_{32} \leq y_2, x_{42} \leq y_2\)
• \(mn\) constraints

• The **polyhedron of an LP** is the feasible region.

• **Definition.** The **LP relaxation** (LPR) of an IP replaces all integer variables \(x_i\) with continuous variables, and non-negativity constraints \(x_i \geq 0\).

• **Definition.** The **polyhedron of an IP** is the feasible region of the LPR.
Valid formulations

- Consider (IP) max \{c^T x: x \in S \subseteq \mathbb{Z}^n\}

- **Defn.** Polyhedron \(P\) is a **valid formulation** for the IP if \(P \cap \mathbb{Z}^n = S\)

---

Strong Formulations

- **Proposition.** Consider two valid formulations \(P_1\) and \(P_2\) for an IP, with \(P_2 \subseteq P_1 \subseteq \mathbb{R}^n\). Then \(z_2^{LP} \leq z_1^{LP}\).

- **Proof.** If \(x \in P_2\) then \(x \in P_1\), and thus \(z_2^{LP} \leq z_1^{LP}\).

- Say that “\(P_2\) is **stronger** than \(P_1\)”
Example

Tight “big M”:
- \( x_{11} + x_{21} + x_{31} + x_{41} \leq 4y_1 \) \( P_1 \)

Loose “big M”:
- \( x_{11} + x_{21} + x_{31} + x_{41} \leq 10y_1 \) \( P_1' \)

- \( P_1' \) allows additional fractional solutions, e.g.
  \( x=(0.5, 0.5, 0.5, 0.5), y=(0.25) \)
- \( P_1 \) is stronger than \( P_1' \)

Importance of Strong Formulations

Strong formulations preclude fractional solutions and improve LP bounding.

Benefit 1:
- More fathoming of open problems by bound.

Benefit 2:
- If “best-bound first,” get better guidance in regard to node selection from LP bound.

Benefit 3:
- (Less important.) Smaller search space.
Convex Hull

- **Definition.** Given set $X \subseteq \mathbb{Z}^n$, the convex hull of $X=\{x^1,\ldots,x^t\}$ is $\text{conv}(X) = \{x: x=\sum_{k=1}^{t} \lambda_k x^k, \sum_{k=1}^{t} \lambda_k = 1, \lambda_k \geq 0 \text{ for all } k\}$

- **Prop.** $\text{conv}(X)$ is a polyhedron.

- **Prop.** Extreme points of $\text{conv}(X)$ all lie in $X$.

- **Prop.** Can solve IP via solving LP on $\text{conv}(X)$.

---

Idea:

Replace IP $\max\{c^T x: x \in X\}$ with LP $\{\max c^T x: x \in \text{conv}(X)\}$
Problem: can require an exponential number of inequalities to describe \( \text{conv}(X) \)

E.g., if problem size has \(|X|=q\) then may need as many as \(2^q\) inequalities!

(Better be the case, else \(P=NP\).)

Complexity theory (brief!)

- **Decision problems**: Is there a TSP tour of cost \(\leq 100\)?
- **P**: class of decision problems that can be easily solved (i.e., in polynomial time)
- **NP**: class of problems for which when answer is YES there is easy (poly time) proof; e.g. TSP tours.
- A problem is **NP-complete** if it is in **NP** and any problem in **NP** can be easily reduced (in poly time) to the problem; e.g. 0/1 integer programs.

It is widely conjectured that \(P \neq NP\)
Ellipsoid and Interior Point methods


• Simplex fast in practice, but not worst-case polynomial time.

• First polynomial-time LP algorithm was devised in 1979 by Khachian, in work that made the headlines.

• Ellipsoid method is an “interior point” algorithm based on specialization of general nonlinear methods. Doesn’t rely on vertex solutions.

• In 1984, Karmarkar announced a polynomial-time interior-point algorithm with solution times 50x better than simplex. Again made headlines!

• LP is in class P, however 0/1 IP is NP-complete.

• If a compact (i.e. polynomially sized) description of conv(X) could be obtained for every IP, then we could solve IPs in polynomial time via reduction to LPs.

• Eureka! We would prove P=NP.
Alternative Goal

• **Q:** What else can we do to improve the strength of formulations?
• **A:** Try to approximate $\text{conv}(X)$ on a given instance in order to automatically strengthen the formulation.

• **Definition.** A **valid inequality** is an inequality that may remove some fractional solutions but removes no integer solutions.
Cuts

• A cut is a valid inequality that removes the optimal fractional solution.

Looking at CPLEX

• IBM’s CPLEX is a commercial solver for integer linear programs
• Routinely used to solve real world problems of large economic significance
• Solver used by AMPL
Some CPLEX features

• Automated Cut Generation
  – At “root” node (Global cuts)
  – At search nodes (Local cuts)

• Automated bound strengthening
  – Tighten right-hand side

• Primal heuristics
  – Try to find integer feasible solutions nearby the current fractional solution at a node

Example

```ampl
var X1 integer >= 0;
var X2 integer >= 0;
maximize Obj: 5 * X1 + 8 * X2;
subject to C1: X1 + X2 <= 6;
subject to C2: 5 * X1 + 9 * X2 <= 45;
end;

ampl: model simple.mod;
ampl: option solver cplex;
ampl: solve;
```
Enabling feedback to AMPL

option cplex_options
    'timing = 1'   # display timing info
    'mipdisplay=2' # display MIP information
    'mipinterval=1';  # node interval

Root relaxation solution time = 0.00 sec. (0.00 ticks)

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Node</th>
<th>Left</th>
<th>Objective</th>
<th>IInf</th>
<th>Best Integer</th>
<th>Best Bound</th>
<th>ItCnt</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>*</td>
<td>0+</td>
<td>41.2500</td>
<td>2</td>
<td>0.0000</td>
<td>70.0000</td>
<td>2</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>41.2500</td>
<td>2</td>
<td>0.0000</td>
<td>41.2500</td>
<td>2</td>
<td>---</td>
</tr>
<tr>
<td>*</td>
<td>0+</td>
<td>0</td>
<td>39.0000</td>
<td>2</td>
<td>41.2500</td>
<td>2 5.77%</td>
<td>2</td>
<td>3.12%</td>
</tr>
<tr>
<td>*</td>
<td>0+</td>
<td>0</td>
<td>40.0000</td>
<td>2</td>
<td>41.2500</td>
<td>2 3.12%</td>
<td>2</td>
<td>0.00%</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>cutoff</td>
<td></td>
<td></td>
<td>40.0000</td>
<td></td>
<td>2</td>
<td>0.00%</td>
</tr>
</tbody>
</table>

Elapsed time = 0.01 sec. (0.03 ticks, tree = 0.00 MB)

Root node processing (before b&c):
    Real time = 0.01 sec. (0.03 ticks)
Parallell b&c, 16 threads:
    Real time = 0.00 sec. (0.00 ticks)
    Sync time (average) = 0.00 sec.
    Wait time (average) = 0.00 sec.
Total (root-branch&cut) = 0.01 sec. (0.03 ticks)

Times (seconds):
    Input = 0.000782
    Solve = 0.018569
    Output = 0.000492
PLEX 12.6.0.0: optimal integer solution; objective 40
2 MIP simplex iterations
0 branch-and-bound nodes
Disabling Presolve

option presolve 0;
option cplex_options
   'timing = 1'
   'mipdisplay=2'
   'mipinterval=1'
   'boundstr=0'
   'dependency=0'
   'coeffreduce=0'
   'presolve=0'
   'cutpass =-1'
   'scale =-1'
   'prerelax = 0'
   'presolvenode = -1';

Disabling Primal Heuristics

option cplex_options
   'fpheur = -1'
   'heurfreq = -1'
   'rinsheur = -1';

fpheur: Whether to use the feasibility pump heuristic on MIP problems (find initial feasible):
-1 = no
  0 = automatic choice (default)
heurfreq: How often to apply "node heuristics" for MIPS
-1= no
  20=every twenty nodes
rinsheur: Relaxation INduced neighborhood Search HEURistic for MIP problems:
-1 = none
  0 = automatic choice of interval (default)
n (for n > 0) = every n nodes.
Summary: BnB II

- **Node Selection**
  - DFS, dual pivots
  - Best-first search

- **Branching decision**
  - Most fractional, pseudo-cost based, user priorities, strong branching

- **Formulation strength**
  - More fathoming, better node selection guidance

- **Cut generation**