IE 511: Integer Programming, Spring 2021 **6 Apr, 2021**

Lecture 21: LP-based Branch and Bound

Lecturer: Karthik Chandrasekaran *Scribe: Karthik*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications.*

We saw an outline of the Branch and Bound technique in the previous lecture. Recall that there are several ways to create a branching tree/enumeration tree for a discrete optimization problem. If we do pure branching, then it leads to a systematic brute-force search of all solutions; moreover, the run-time is dominated by the number of nodes in the enumeration tree which could be exponential. So, we would like to prune the tree cleverly to rule out branches that need not be explored. We saw pruning rules based on bounds in the previous lecture. Bounds can be derived by several means—one common way to derive bounds is by solving the LP-relaxation. To illustrate this, we will solve an example via LP-based Branch and Bound today. Subsequently, we will address the considerations that should be taken into account while implementing LP-based Branch and Bound.

21.1 LP-based B&B

A popular way to obtain upper bound for a maximization problem is by solving the LP-relaxation of the problem. This can be used to obtain bounds and implement the B&B approach. We will illustrate LP-based B&B through an example and then touch upon implementation considerations.

21.1.1 An example

$$\text{Consider } z = \max\{4x_1 - x_2 : x \in P \cap \mathbb{Z}^5\}$$

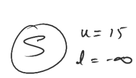
$$\text{where } P := \left\{ \begin{array}{ll} x \in \mathbb{R}^5 : & 3x_1 - 2x_2 + x_3 = 14 \\ & x_2 + x_4 = 3 \\ & 2x_1 - 2x_2 + x_5 = 3 \\ & x \geq 0 \end{array} \right\}.$$

$$\text{Let } S := P \cap \mathbb{Z}^5.$$

Bounds(S):

1. Upper bound: Solve the LP-relaxation. An optimal solution is $\tilde{x} = \left(\frac{9}{2}, 3, \frac{13}{2}, 0, 0\right)$. Therefore, $u = 15$.

2. Lower Bound: Lower bound for a maximization problem is given by a feasible solution. If we do not have a feasible solution yet, then $l = -\infty$ is a conventional lower bound. Therefore, the tree looks as below.



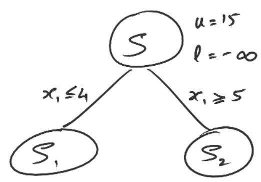
Since $l < u$, we do not have optimality/infeasibility so we need to branch, i.e., split the feasible region.

Branch:

Common way to split is to use an integer variable that is taking a fractional value in the current LP solution. In this example, $\tilde{x}_1$ fractional, so let us branch on this variable. Let

$$S_1 := \{x \in S : x_1 \leq \lfloor \tilde{x_1} \rfloor\},$$
$$S_2 := \{x \in S : x_1 \leq \lceil \tilde{x_1} \rceil\}.$$

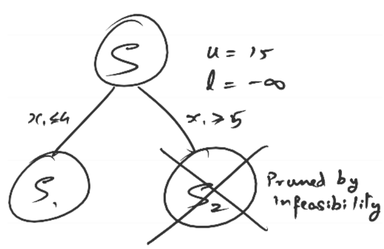
Note that $S = S_1 \cup S_2$ and $S_1 \cap S_2 = \emptyset$ and $\tilde{x}$ is infeasible for both S_1 and S_2.



Which node to explore next? i.e., which subproblem to solve next?

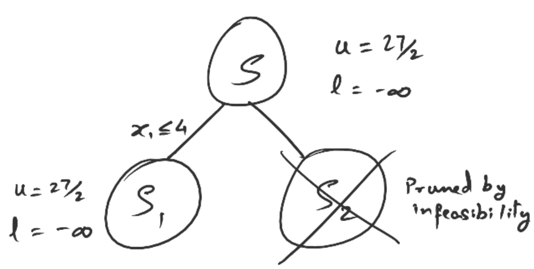
Choosing an active node: Say we explore S_2.

Bounds(S_2): Solve $u^2 := \max\{4x_1 - x_2 : x \in P_2\}$ where $P_2 = \{x \in P : x_1 \geq 5\}$. It is infeasible. So, we can prune S_2.



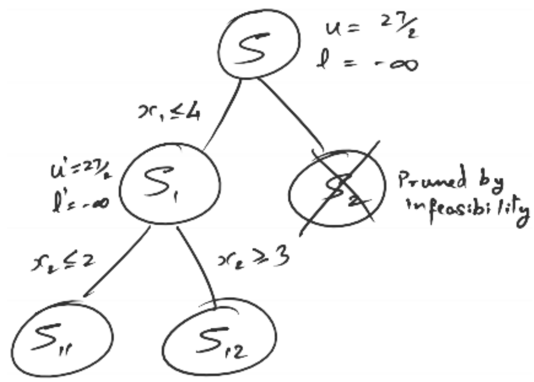
Choose an active node: S_1

Bounds(S_1): Solve $u^1 := \max\{4x_1 - x_2 : x \in P_1\}$ where $P_1 := \{x \in P : x \leq 4\}$. An optimal solution is $\tilde{x} = \left(4, \frac{5}{2}, 7, 4, 0\right)$. Therefore, $u^1 = \frac{27}{2}$. We note that $u^1 \leq u$, so we update the bounds.



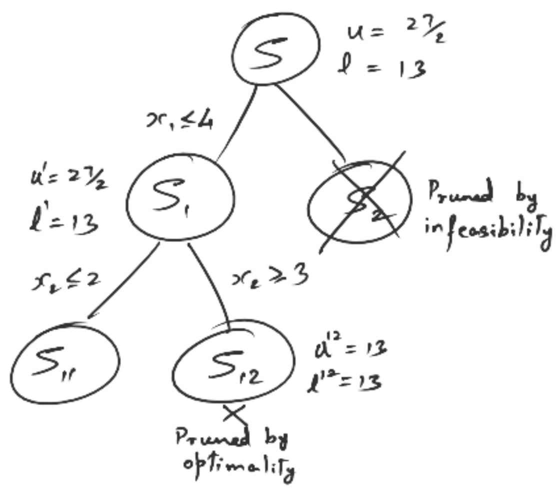
Branch: $S_{11} := \{x \in S_1 : x_2 \leq 2\}$ and $S_{12} := \{x \in S_1 : x_2 \geq 3\}$.

Note that again $S_1 = S_{11} \cup S_{12}$ and $S_{11} \cap S_{12} = \emptyset$ and $\tilde{x}$ from S_1 is infeasible for S_{11} and S_{12}.



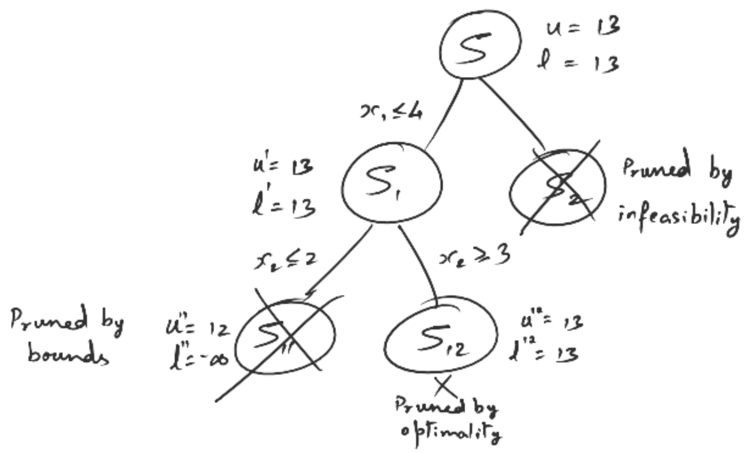
Choose an active node: S_{12}

Bounds(S_{12}): Solve $u^{12} := \max\{4x_1 - x_2 : x \in P_{12}\}$ where $P_{12} := \{x \in P_1 : x_2 \geq 3\}$. An optimal solution is $\tilde{x} = (4, 3, 8, 1, 0)$ which implies that $u^{12} = 13$ and $l^{12} = 13$. We can prune S_{12} by optimality and update the bounds.



Choose an active node: S_{11}

Bounds(S_{11}): Solve $u^{11} := \max\{4x_1 - x_2 : x \in P_{11}\}$ where $P_{11} := \{x \in P_1 : x_2 \leq 2\}$. An optimal solution is $\tilde{x} = \left(\frac{7}{2}, 2, \frac{15}{2}, 1, 0\right)$ which implies that $u^{11} = 12$. We can prune S_{11} by bound and update the bounds.



Therefore, the optimal objective value is 13 and an optimal solution is $(4, 3, 8, 1, 0)$.

21.1.2 Implementation Considerations in LP-based B&B

While implementing LP-based B&B, certain careful considerations tends to speed up computations.

1) **Re-optimizing:** LP(S) and LP(S_1) differ only in one constraint. To optimize for LP(S_1), start from the optimal solution for LP(S) and use dual simplex—this tends to be fast.

2) **Storing the tree:** Instead of storing all nodes of the tree, it suffices to store only active nodes.

3) **Bounding:** To obtain upper bounds, solve the LP-relaxation. To obtain lower bounds, use heuristics and approximation algorithms.

4) **Branching:** Branch on a variable that is fractional in the optimal solution to the LP-relaxation. If there are several such variables, use the *most fractional* one. There are numerous other branching rules based on estimating the *cost of a variable to become an integer*.

5) **Choosing an active node to explore:** We chose arbitrarily in the example.

   - Depth first strategy:
     - *Intuition:* Tree can be pruned significantly only if there is a good feasible solution which gives a good lower bound.
     - *Advantage:* Can reoptimize fast using dual simplex.
   - Breadth first strategy:
     - *Intuition:* We would like to minimize the number of explored nodes so choose an active node with the best upper bound.
   - In practice, a combination of both is employed: Initially, follow the depth first strategy to obtain a good feasible solution. Subsequently, do a mix of both strategies.

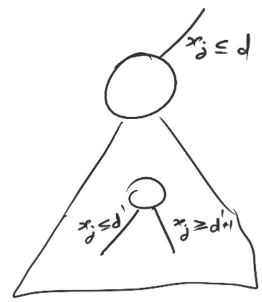
21.1.3 LP-based B&B: Finiteness

Next, we will address the question of whether LP-based B&B will terminate in finite time. We will answer this affirmatively by showing that the enumeration tree in LP-based B&B is finite.

Lemma 0.1. *Suppose $P = \{x \in \mathbb{R}^n : Ax \le b, x \ge 0\}$ is bounded. Then the enumeration tree in an LP-based B&B will be finite. In particular, if $w_j := \lceil \max\{x_j : x \in P\} \rceil$ then every path in the tree can contain at most $\sum_{j=1}^{n} w_j$ nodes. Therefore,*

$$\textit{Depth of tree } \le \sum_{j=1}^{n} w_j.$$

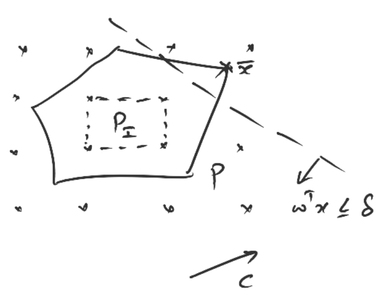
Proof. After adding the constraint $x_j \le d$ the only other constraint for x_j that can subsequently appear are $x_j \le d'$ and $x_j \ge d'+1$ for some $d' \in \{0, \ldots, d-1\}$ (see Figure below). Therefore, the largest number of branches involving x_j is at most w_j. $\square$



21.2 Cutting Plane Approach

Next, we will see another solving technique for unstructured IPs, namely the cutting plane approach. Consider $z = \max\{c^T x : x \in P \cap \mathbb{Z}^n\}$ where P is a given polyhedron (it is given by specifying the constraint matrix A and RHS vector b such that $P = \{x : Ax \le b\}$). Recall that $P_I =$ convex-hull$(P \cap \mathbb{Z}^n)$ is a polyhedron and also that it is sufficient to solve $\max\{c^T x : x \in P_I\}$.

We first describe the geometric viewpoint of the cutting plane procedure. Let $\bar{x} = \arg\max\{c^T x : x \in P\}$ be an extreme point optimum. If $\bar{x}$ is integral, then it is an optimum to the IP and we are done. Otherwise, we find an inequality $w^T x \le \delta$ which is valid for P_I but violated by $\bar{x}$; next, we re-solve the LP $\max\{c^T x : x \in P, w^T x \le \delta\}$, and repeat (see figure below).



A formal description of the cutting plane algorithmic approach is given in Algorithm 1. Note that it is not a complete algorithm since it does not explicitly mention how to find a valid inequality for

P_I that is violated by $\bar{x}$.

Algorithm 1: Cutting Plane Algorithmic Approach

Input: P and c
Output: $\bar{x} = \arg\max\{c^T x : x \in P \cap \mathbb{Z}^n\}$

Initialize $Q = P$
repeat
    Solve $\max\{c^T x : x \in Q\}$ to find an extreme point optimum $\bar{x}$
    if $\bar{x} \in \mathbb{Z}^n$ **then**
        STOP and return $\bar{x}$
    else
        find a valid inequality $w^T x \leq \delta$ for P_I such that $w^T \bar{x} > \delta$
        $Q \leftarrow Q \cap \{x : w^T x \leq \delta\}$

Remark. It is important that the optimum $\bar{x}$ for $\max\{c^T x : x \in Q\}$ that we find in Algorithm 1 is an extreme point optimum. Only then can we find a valid inequality for P_I that is violated by $\bar{x}$. See the geometric viewpoint and the figure therein.

Issues. We have to address two issues pertaining to Algorithm 1:

1. How to find a valid inequality $w^T x \leq \delta$ for P_I that is violated by $\bar{x}$?
2. Even if we can find such inequalities, will the algorithm terminate in finite time?