Strengthening IP formulations; The Branch and Cut Algorithm

Milind G. Sohoni[1]

January 3, 2015

[1] Indian School of Business, Hyderabad, India, e-mail: milind_sohoni@isb.edu

Strengthening Formulations

- Consider two formulations A and B of the same ILP
- Let P_A and P_B denote the LP relaxations of these formulations respectively
- Formulation A is said to be *at least as strong as* B if $P_A \subseteq P_B$
- If the inclusion is "strict" then A is *stronger* than B

Strengthening Formulations

- Often, a given formulation can be strengthened with additional inequalities satisfied by all feasible integer solutions
- Consider the following example: *The Perfect Matching Problem*
  - We are given a set of n people that need to be paired in teams of two
  - Let c_{ij} represent the cost of pairing person i with person j
  - Our goal is to minimize the overall cost across all pairings
  - We can represent this problem on a graph $G = (N, E)$ where the nodes N represent people and the edges E represent all possible pairings

The Perfect Matching Formulation

We have $x_{ij} = 1$ if the endpoints i and j are "matched", and $x_{ij} = 0$ otherwise.

$$\begin{array}{lll} \min & \displaystyle\sum_{\{i,j\}\in E} c_{ij}x_{ij} & \\ \text{s.t.} & \displaystyle\sum_{\{j \mid \{i,j\}\in E\}} x_{ij} = 1 & \forall\ i \in N \\ & x_{ij} \in \{0,1\} & \forall\ \{i,j\} \in E. \end{array}$$

Valid Inequalities and Cutting Planes

- Suppose we formulate it as an integer program by specifying a rational polyhedron $P = \{x \in \mathbb{R}^n_+ | \; Ax \leq b\}$such that $S = Z^n \cap P$
- Hence $S = \{x \in \mathbb{Z}^n_+ | \; Ax \leq b\}$and $conv(S)$ is the convex hull of S i.e., the set of points that are convex combinations of points in S
  - $conv(S) \subseteq S$; "ideal" if $conv(S) = S$
- An inequality $\pi^T x \leq \pi_0$ is called a *valid inequality* if it is satified by all points in S
- *Cutting planes*: Given a formulation for S identify additional "valid inequalities (constraints)" that remove regions of S that contain no feasible solutions – thus obtaining a "better" formulation for S

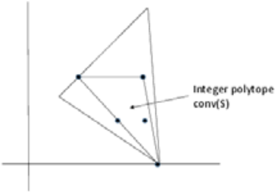

Figure: Convex hull of an integer program.

Example of Valid Inequalities

- Suppose
  $S = \left\{x \in \{0,1\}^5 \mid 3x_1 - 4x_2 + 2x_3 - 3x_4 + x_5 \leq -2\right\}$
- Observe that $x_2 = x_4 = 0$, then $3x_1 + 2x_3 + x_5 \leq -2$ is impossible! So, $x_2 + x_4 \geq 1$ must be a valid inequality
- Similarly, if $x_1 = 1$ and $x_2 = 0$, then
  $0 \leq 3 + 2x_3 - 3x_4 + x_5 \leq -2$ is again impossible! So, $x_1 \leq x_2$ is also a valid inequality

Back to the Pefect Matching Problem

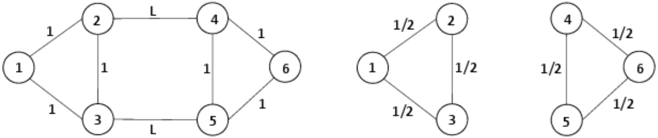

Figure: Valid Inequalities for the Perfect Matching Problem.

- Consider the graph on the left
- The *optimal perfect matching* has a value $L + 2$
- The optimal solution to the LP relaxation has a value 3
- The formulation can be extremely "weak"
- Add the valid inequality $x_{24} + x_{35} \geq 1$; Every perfect matching satifies this inequality

The Odd Set Inequalities

- We can generalize the inequality from the previous slide
- Consider a "cut" S corresponding to any odd set of nodes (a cut separates the nodes into two sets)
- The "cutset" corresponding to S is

$$\delta(S) = \{\{i,j\} \in E \mid i \in S,\, j \notin S\}$$

- An "odd cutset" is any $\delta(S)$ for which $|S|$ is odd
- Note that every perfect matching contains at least one edge from every odd cutset
- Hence each odd cutset induces a possible valid inequality

$$\sum_{\{i,j\} \in \delta(S)} x_{ij} \geq 1, \;\; S \subset N, \;\; \text{and } |S| \text{ odd}$$

Generating Constraints

- If we add all of the odd set inequalities, the new formulation would be “ideal”
- However, the number of inequalities (for a general problem) could be exponential in size
- Only a few of these inequalities will eventually be “active” in the optimal solution
- Essentially, we generate these constraints *on the fly*
  - Solve the initial LP relaxation
  - If solution is feasible, STOP; Else look for a violated odd set inequality
  - Add the inequality and reoptimize; Go to the earlier step

Branch and Cut Algorithms

- If we combine constraint generation with the Branch and Bound algorithm, we get the *Branch and Cut* algorithm
- The relaxation at each node is “strengthened” using valid inequalities
- This increases the lower bound and improves efficiency
- Most state of art IP solvers use *Branch and Cut*