Practical guidelines for solving difficult mixed integer linear programs

Klots and Newman

Fundamentals (2)

- Branch and bound
  - Search tree, where linear relaxation problems are solved at each node for a dual bound. We pick a variable $x_j = f$, and branch on $x_j \leq \lfloor f \rfloor$ and $x_j \geq \lceil f \rceil$.
  - Nodes can be pruned if infeasible or if node dual bound is worse than the current incumbent integer solution.
  - Since the branch and bound tree grows exponentially, the ability to prune elements is important
  - Optimality gap
  - Branch and cut to iteratively add constraints (cuts) as long as integer solutions aren't removed from the set of feasible solutions.
  - Algorithm uses the difference between the dual and primal bound to determine quality relative to optimality (or will halt when every node has been processed)

Troubles and remedies (3, 3.1, 3.2)

Lack of node throughput

Each node of B&B-tree has an LP problem. Low node throughput is when the amount of iterations, required to solve these subproblems, are high.

Remedies: Consider Primal vs. Dual.

Tune appropriate alg. parameters.

Lack of progress in the best integer solution

Remedies: Provide initial feasible solution, even though it might seem trivial.

- Obvious solution
- Solve related auxiliary problem
- Use solution from prev. iter. for sequence of models

Depth first search in B&B

Lack of progress in best bound (3.3-3.4)

- Branch & Bound may have trouble obtaining good bounds via LP relaxations
  - Bounds can be strengthened by modifying the LP relaxation using *cuts*
    - Many types of cuts exist
  - If cuts does not prove useful, many modern optimizers have further parameter settings that can help pruning further or strengthening the formulation
- Search strategy
    - *Best Bound node selection*: Select the node with the lowest objective value in the LP relaxation.
      - Not guaranteed to find an *integer feasible* solution faster, so buyer beware
    - *Strong branching*: Use dual problems and infeasible branches to tighten variable bounds
    - *Probing*: Fix binary variables and propagate to other variables through intersecting constraints
      - Helps the optimizer find variables that can *always* only assume the same value in any feasible solution
    - *More aggressive cut generation*: Well, more cuts
- The optimizer solves linear programs at each node of branch-and-bound tree, so the practitioner must be careful to avoid the numerical performance issues.
- It is important to avoid large differences in orders of magnitude in data to preclude the introduction of unnecessary round-off error
- Differences of input values create round-off error in floating point calculations which makes it difficult for the algorithm to distinguish between this error and a legitimate value.

Types of cuts for B&B

Table 1
Different types of cuts and their characteristics, where z is binary unless otherwise noted, and x is continuous.

Cut name	Mathematical description of cut	Structure of original MILP that generates the cut
Clique[b]	$\sum_i z_i \leq 1$	Packing constraints
Cover[b]	$\sum_i z_i \leq b$, b integer	Knapsack constraints
Disjunctive[a]	Constraint derived from an LP solution	$\sum_i a'_i x_i \geq b'$ or $\sum_i a''_i x_i \geq b''$, x_i continuous or integer
Mixed Integer Rounding[a]	Use of floors and ceilings of coefficients and integrality of original variables	$a_C x_C + a_I x_I = b, x \geq 0$
Generalized Upper Bound[b]	$\sum_i x_i \leq b$, b integer	Knapsack constraints with precedence or packing
Implied Bound[b]	$x_i \leq \frac{b}{a_i}$	$\sum_i a_i x_i \leq bz, x \geq 0$
Gomory[a]	Mixed integer rounding applied to a simplex tableau row $\bar{a}$ associated with optimal node LP basis	$\bar{a}_C x_C + \bar{a}_{I/k} x_{I/k} + x_k = \bar{b}$, x_k integer, $x \geq 0$
Zero-half[a]	$\lambda^T A x \leq \lfloor \lambda^T b \rfloor$, $\lambda_i \in \{0, 1/2\}$	Constraints containing integer variables and coefficients
Flow Cover[b]	Linear combination of flow and binary variables involving a single node	Fixed charge network
Flow Path[b]	Linear combination of flow and binary variables involving a path of nodes	Fixed charge network
Multicommodity flow[b]	Linear combination of flow and binary variables involving nodes in a network cut	Fixed charge network

[a] Based on general polyhedral theory.
[b] Based on specific, commonly occurring problem structure.

Tighter formulations (4)

Why is a problem difficult to solve? What can we do?

1. Simplify the model if necessary
   - Remove/group constraints or
2. Identify the constraints that prevent the objective function from improving
   - Locate constraints that prevent the trivial solution from being possible
3. Focus on cuts that actually tighten the problem

Tighter formulations (4)

1. Linear or logical combinations of constraints
2. Optimize one or more related models
3. Use of the incumbent solution objective value
4. Disjunctions
   Suppose $X_1 = \{x : a^T x \geq b\}$ $X_2 = \{x : \hat{a}^T x \geq \hat{b}\}$.
   Componentwise: $u_j = \max\{a_j, \hat{a}_j\}$ *and* $\bar{u} = \min\{b, \hat{b}\}$.
   Then $u^T x \geq \bar{u}$
   This is valid for the union of X_1 and X_2, and thereby also valid for their convex hull.
5. The exploitation of infeasibility
   Infeasible: $a^T x \leq b$
   Then Valid cut: $a^T x \geq b + 1$

Table 2
Under various circumstances, different formulations and algorithmic settings have a greater chance of faster solution time on an integer programming problem instance.

Characteristic	Recognition	Suggested tactic(s)
• Troublesome LPs	• Large iteration counts per node, especially regarding root node solve	• Switch algorithms between primal and dual simplex; if advanced starts do not help simplex, consider barrier method
• Lack of progress in best integer	• Little or no change in best integer solution in log after hundreds of nodes	• Use best estimate or depth-first search • Apply heuristics more frequently • Supply an initial solution • Apply discount factors in the objective • Branch up or down to resolve integer infeasibilities
• Lack of progress in best node	• Little or no change in best node in log after hundreds of nodes	• Use breadth-first search • Use aggressive probing • Use aggressive algorithmic cut generation • Apply strong branching • Derive cuts *a priori* • Reformulate with different variables
• Data and memory problems	• Slow progress in node solves • Out of memory error	• Avoid large differences in size of data • Reformulate "big M" constraints • Rectify LP problems, e.g., degeneracy • Apply memory emphasis setting • Buy more memory

Tighter formulations: Example 1

Mixed integer rounding cut:

$$4x_1 + 3x_2 + 5x_3 = 10 \qquad (12)$$

$$x_1, x_2, x_3 \geq 0, \quad \text{integer.} \qquad (13)$$

Divide by 4

$$x_1 + \frac{3}{4}x_2 + \frac{5}{4}x_3 = \frac{5}{2} \qquad (14)$$

Split into integer and fractional:

$$\underbrace{x_1 + x_2 + x_3}_{\hat{x}} - \frac{1}{4}x_2 + \frac{1}{4}x_3 = 2 + \frac{1}{2} = 3 - \frac{1}{2} \quad (15)$$

Consider: $\hat{x} \leq 2$

$$\hat{x} \leq 2 \Rightarrow \frac{-1}{4}x_2 + \frac{1}{4}x_3 \geq \frac{1}{2} \Rightarrow x_3 \geq 2 \quad (16)$$

Consider: $\hat{x} > 3$

$$\hat{x} \geq 3 \Rightarrow \frac{-1}{4}x_2 + \frac{1}{4}x_3 \leq \frac{-1}{2} \Rightarrow x_2 \geq 2 \quad (17)$$

This gives the new constraint:

$$x_2 + x_3 \geq 2 \qquad (18)$$

Tighter formulations: Example 2

Consider the following one-constraint system:

$$13429x_1 + 26850x_2 + 26855x_3 + 40280x_4 + 40281x_5$$
$$+ 53711x_6 + 53714x_7 + 67141x_8 = 45094583$$
$$x_j \geq 0, \quad \text{integer}, j = 1, \ldots, 8.$$

All variables have coefficients close to multiples of 13429

$$13429 \underbrace{(x_1 + 2x_2 + 2x_3 + 3x_4 + 3x_5 + 4x_6 + 4x_7 + 5x_8)}_{\hat{x}} \tag{19}$$

$$-8x_2 - 3x_3 - 7x_4 - 6x_5 - 5x_6 - 2x_7 - 4x_8 \tag{20}$$
$$= 3358 * 13429 + 1 = 3359 * 13429 - 13428. \tag{21}$$

We see that if $\hat{x} \leq 3358$, the model is infeasible since:

$$\Rightarrow \underbrace{-8x_2 - 3x_3 - 7x_4 - 6x_5 - 5x_6 - 2x_7 - 4x_8}_{\leq 0} \geq 1.$$

Therefore $\hat{x} \geq 3359$ is a valid cut and we can add it to the model as a constraint.

$$x_1 + 2x_2 + 2x_3 + 3x_4 + 3x_5 + 4x_6 + 4x_7 + 5x_8 \geq 3359 \tag{24}$$
$$8x_2 + 3x_3 + 7x_4 + 6x_5 + 5x_6 + 2x_7 + 4x_8 \geq 13428. \tag{25}$$

This allows CPLEX to identify the system as infeasible without much computation, see (Node Log #8).

More examples

Original problem:

$$(MIQP) \ \max \sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij} x_i x_j$$

$$\text{subject to} \sum_{j=1}^{n} x_j \leq k$$

x_j binary.

Rewritten:

$$(MILP) \max \sum_{j=1}^{n} \sum_{\substack{i=1 \\ i<j}}^{n} d_{ij} z_{ij}$$

$$\text{subject to} \sum_{j=1}^{n} x_j \leq k$$

$$z_{ij} \leq x_i \quad \forall\, i,j$$

$$z_{ij} \leq x_j \quad \forall\, i,j$$

$$x_i + x_j \leq 1 + z_{ij} \quad \forall\, i,j$$

$$x_j, z_{ij} \text{ binary} \quad \forall\, i,j.$$

This linearized model instance with n = 60 and k = 24 possesses 1830 binary variables and 5311 constraints.

- Ran out of memory after 4 hours.

If z_{ij} = 1 then x_i = 1 = x_j

If 0<=z_{ij}<1 then x_i or x_j must be zero in the MILP, but not in the relaxation.

- In the LP relaxation, we can set more of the z variables to positive values than in the MILP

Globally valid cut:

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} z_{ij} \leq k(k-1)/2$$

- Solved in 2.5 hours with cut