SE524/EC524 Optimization Theory and Methods

Yannis Paschalidis
yannisp@bu.edu, http://ionia.bu.edu/

Department of Electrical and Computer Engineering,
Division of Systems Engineering,
and Center for Information and Systems Engineering,
Boston University

Lecture 22: Outline

1. Families of Integer Programming methods.
2. Cutting plane methods.
3. Branch and bound.
4. Dynamic programming: Traveling salesman problem.
5. IP duality.
6. Approximation algorithms.
7. Heuristics.

Integer Programming methods

1. **Exact Methods** (EXP-time): cutting plane, branch and bound, DP.
2. **Approximation Methods** (POLY-time): suboptimal solutions.
3. **Heuristic Methods**: local search, simulated annealing.

Cutting Plane methods

$$\begin{aligned} \min \quad & \mathbf{c}'\mathbf{x} \qquad \textbf{(IP)} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \\ & \mathbf{x} : \text{integer} \end{aligned}$$

Cutting plane algorithm:

1. Solve LP relaxation of IP$\Rightarrow \mathbf{x}^*$.
2. If $\mathbf{x}^*$ integer STOP.
3. Otherwise "cut" $\mathbf{x}^*$: add violating constraint which is satisfied by all IP feasible solutions.

How do we cut ? **Gomory cuts**.

Branch and Bound

- Solve LP relaxation$\Rightarrow$ $\mathbf{x}^*$. Select fractional component x_i^*.
- Create two subproblems F_1, F_2 by adding either of:

$$x_i \leq \lfloor x_i^* \rfloor \qquad \text{or} \qquad x_i \geq \lceil x_i^* \rceil$$

  Make both **active**.
- LP relaxation of each subproblem F_i yields lower bound $b(F_i)$.
- Maintain upper bound U on IP optimal cost (U obtained by evaluating the cost of an IP feasible sol., e.g., when a subproblem has an integer optimal sol.)

Branch and Bound algorithm

1. Initialize $U = \infty$.
2. Select an active subproblem.
3. Consider LP relaxation. If infeasible delete subproblem. Else solve.
4. If optimal solution integer delete subproblem and update U. Else compute $b(F_i)$.
5. If $b(F_i) \geq U$, delete subproblem.
6. If $b(F_i) < U$ divide into two further subproblems.

Approximation algorithms

Definition

For a minimization problem: An algorithm is an ϵ-approximation algorithm if it runs in POLY-time and returns a feasible solution Z such that

$$Z \leq (1+\epsilon)Z^*$$

For maximization:

$$Z \geq (1-\epsilon)Z^*$$