Parametric Programming

Let $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ and $c, d \in \mathbb{R}^n$.

Parametric Program.

Solve, for all $\theta \in \mathbb{R}$:

$$\begin{aligned} g(\theta) := \min \ & (c + \theta \cdot d)^T \cdot x \\ \text{s.t.} \ & A \cdot x = b \\ & x \geq 0 \end{aligned}$$

Assume that $\{x \mid A \cdot x = b,\, x \geq 0\} \neq \emptyset$. Then

$$g(\theta) = \min_{i=1,\ldots,N} (c + \theta \cdot d)^T \cdot x^i$$

for those θ with $g(\theta) > -\infty$ where $x^1, \ldots, x^N$ are the extreme points.

Parametric Programming: Example

$$\begin{array}{lrcrcrcl} \min & (-3 + 2\theta)\, x_1 & + & (3 - \theta)\, x_2 & + & x_3 & & \\ \text{s.t.} & x_1 & + & 2\, x_2 & - & 3\, x_3 & \leq & 5 \\ & 2\, x_1 & + & x_2 & - & 4\, x_3 & \leq & 7 \\ & & & & & x_1,\, x_2,\, x_3 & \geq & 0 \end{array}$$

Chapter 11:
Large-Scale Linear Programmming

(cp. Bertsimas & Tsitsiklis, Chapter 6)

Delayed Column Generation

Let $A \in \mathbb{R}^{m \times n}$ with $\text{rank}(A) = m$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and $m \ll n$.

$$\begin{aligned} \min \;\; & c^T \cdot x \\ \text{s.t.} \quad & A \cdot x = b \\ & x \geq 0 \end{aligned}$$

Suppose that the number of columns n is huge such that A cannot be generated and stored in your computer's memory.

Remember: Revised simplex method only requires m basic columns and the column which shall enter the basis.

Pricing problem: How to find column that should enter basis (i. e., $\bar{c}_j < 0$)?

Solution: Sometimes one can find j with $\bar{c}_j = \min_i \bar{c}_i$ efficiently.

Conclusion:

- Only work with few columns at a time (basic columns and some "promising" non-basic columns).
- Generate new relevant columns by solving pricing problem.

Example: Min-Cost Multi-Commodity Flows

Given: Digraph $D = (V, A)$, capacities $u : A \to \mathbb{R}_{\geq 0}$, costs $c : A \to \mathbb{R}_{\geq 0}$; k source-sink pairs $(s_i, t_i) \in V \times V$ with demands $d_i \in \mathbb{R}_{\geq 0}$, $i = 1, \dots, k$.

Task: Send d_i units of flow from s_i to t_i for all i without violating arc capacities; minimize total cost.

Path-based LP formulation: Let $\mathcal{P}_i$ be the set of all s_i-t_i-dipath in D, $\mathcal{P} := \bigcup_{i=1}^k \mathcal{P}_i$. Cost of path $P \in \mathcal{P}$ is $c_P := \sum_{a \in P} c(a)$.

$$\begin{aligned}
\min \quad & \sum_{P \in \mathcal{P}} c_P \, x_P \\
\text{s.t.} \quad & \sum_{P \in \mathcal{P}: a \in P} x_P + s_a = u(a) && \text{for all } a \in A \\
& \sum_{P \in \mathcal{P}_i} x_P = d_i && \text{for all } i = 1, \dots, k \\
& x_P, s_a \geq 0 && \text{for all } P \in \mathcal{P},\ a \in A
\end{aligned}$$

Notice: The number of variables is exponential in the size of D.

Pricing Problem and Dual Separation Problem

Consider the dual LP:

$$\begin{aligned}
\max \quad & \sum_{a \in A} u(a) \cdot y_a + \sum_{i=1}^{k} d_i \cdot z_i \\
\text{s.t.} \quad & z_i + \sum_{a \in P} y_a \leq c_P && \text{for all } P \in \mathcal{P}_i,\ i = 1, \dots, k \\
& y_a \leq 0 && \text{for all } a \in A
\end{aligned}$$

Notice: The reduced cost of a primal variable is negative if and only if the corresponding dual constraint is violated ($\longrightarrow$ dual separation problem).

Easy for slack variable s_a: Check whether $y_a > 0$.

For path variable x_P, $P \in \mathcal{P}_i$: $\quad z_i + \sum_{a \in P} y_a > c_P = \sum_{a \in P} c(a)$

$$\iff \quad \sum_{a \in P} (c(a) - y_a) < z_i$$

Conclusion: Solve pricing problem by computing shortest s_i-t_i-paths w.r.t. arc weights $c(a) - y_a$, for $i = 1, \dots, k$.

Cutting Plane Methods

Delayed column generation viewed in terms of the dual LP:

$$\max \quad p^T \cdot b \qquad \text{s.t.} \quad p^T \cdot A_i \leq c_i \qquad \text{for all } i = 1, \ldots, n$$

If n is huge, instead of dealing with all n constraints, restrict to subset $I \subset \{1, \ldots, n\}$ and consider relaxed problem

$$\max \quad p^T \cdot b \qquad \text{s.t.} \quad p^T \cdot A_i \leq c_i \qquad \text{for all } i \in I$$

Let p^* be an optimal basic feasible solution:

- If p^* is feasible for original LP, it is also optimal there.
- Otherwise, find a violated constraint and add it to relaxed problem.

Remark: Notice the similarity to the ellipsoid method where, in every iteration, the separation problem needs to be solved.

Example: Solving the Subtour LP

For a given TSP instance, consider the subtour LP:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(v)} x_e = 2 && \text{for all nodes } v \in V, \\ & \sum_{e \in \delta(X)} x_e \geq 2 && \text{for all subsets } \emptyset \neq X \subsetneq V, \qquad (*) \\ & 0 \leq x_e \leq 1 && \text{for all edges } e. \end{aligned}$$

Notice that there are $2^{n-1} - 1$ subtour elimination constraints (*).

The corresponding separation problem is a min-cut problem that can be solved efficiently by network flow methods.

Conclusion: Subtour LP is typically being solved by cutting plane methods.