Lecture 7 - Column Generation

Lan Peng, Ph.D.

School of Management, Shanghai University, Shanghai, China

"Stepping bravely into the unknown."

1 Fundamental Idea of the Column Generation

We first introduce an intuitive approach based on the Revised Simplex Method. Recall the Minkowski-Weyl Theorem

Theorem 1.1 (Minkowski-Weyl Theorem)**.** *Let $P = \{x \in \mathbb{R}^n | Ax \leq b\}$ be a polyhedron with extreme points $V(P) = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_{|V|}\}$ and extreme directions $R(P) = \{\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_{|R|}\}$, then*

$$\begin{aligned} P = \{x \in \mathbb{R}^n | x = & \sum_{j=1}^{|V|} \lambda_j \mathbf{v}_j + \sum_{j=1}^{|R|} \mu_j \mathbf{r}_j \\ & \sum_{j=1}^{|V|} \lambda_j = 1, \lambda_j \geq 0, \quad \forall j = 1, 2, \cdots, |V| \\ & \mu_j \geq 0, \quad \forall j = 1, 2, \cdots, |R|\} \end{aligned}$$

With out lost of generality, a (bounded) linear problem model can be formulated as follows

$$\begin{aligned} \text{(MP)} \quad z^*_{MP} = \min \quad & \sum_{j \in J} c_j \lambda_j \\ \text{s.t.} \quad & \sum_{j \in J} \mathbf{a}_j \lambda_j \geq \mathbf{b} \quad [\pi] \\ & \lambda_j \geq 0, \quad \forall j \in J \end{aligned}$$

Where each λ_j represents an extreme point of the polyhedron. We call this formulation the master problem (MP). The Revise Simplex Method allows us to solve the LP with a **large** $|J|$, which is, in each iteration add one more nonbasic variable with negative reduce cost into the formulation. Since we do not need to consider all variables in the beginning, we can start with solving a restricted version of the master problem with only a subset of J (in fact, we can start the iteration with a B^{-1}, which is any basic feasible solution). That is

$$\begin{aligned} \text{(RMP)} \quad z^*_{RMP} = \min \quad & \sum_{j \in J'} c_j \lambda_j \\ \text{s.t.} \quad & \sum_{j \in J'} \mathbf{a}_j \lambda_j \geq \mathbf{b} \quad [\pi] \\ & \lambda_j \geq 0, \quad \forall j \in J' \subset J \end{aligned}$$

We call this formulation the restricted master problem (RMP). In the RMP, we have less choices than the MP, therefore, as a minimization problem,

$$z^*_{RMP} \geq z^*_{MP}$$

provides an upper bound for MP. In each iteration, a new variable - column - is introduced into the formulation, thus, this approach is called the Column Generation.

Algorithm 1 Column generation with explicit pricing

Initialize with a set $J' \subseteq J$ of variables
repeat
  Solve RMP to optimality, obtain λ and π
  Calculate $\bar{c}_j = z_j - c_j = \pi^\top \mathbf{a_j} - c_j, \forall j \in J$
  if $\exists \lambda_{j^*}$ with $z_{j^*} - c_{j^*} < 0$ **then**
    $J' \leftarrow J' \cup \{j^*\}$
until $\forall \lambda_j, j \in J, z_j - c_j \geq 0$

Notice that if $|J|$ is **huge**, it will be just too costly to calculate the reduce cost for *every* nonbasic variable, sometimes (in fact, most of the times) we are not even able to enumerate all nonbasic variables, as the size of J is usually exponentially large. Thus, an optimization problem is introduced to find the least-reduce-cost nonbasic variable, to replace the explicit pricing of all nonbasic variables, we call it the *pricing subproblem*. The updated Column Generation algorithm is as follows

Algorithm 2 Column generation with pricing subproblem

Initialize with a set $J' \subseteq J$ of variables
repeat
  Solve RMP to optimality, obtain λ and π
  Solve $\bar{c}_{j^*} = \min\{z_j - c_j = \pi^\top \mathbf{a_j} - c_j, \forall j \in J\}$
  if $\bar{c}_{j^*} < 0$ **then**
    $J' \leftarrow J' \cup \{j^*\}$
until $\bar{c}_{j^*} \geq 0$

Takeaway: explicit enumeration of all patterns is totally out of the question!! The difficulty lies in defining the pricing problem.

2 The Dantzig-Wolfe Reformulation

The previous section does not distinguish the "easy" part and the "difficult" part of the constraints, however, in real problems, some constraints might be easier to dealt with, while others are more complicated. We can equivalently reformulate the LP model as

$$\begin{aligned} (LP) \quad \min \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \quad (\text{"difficult" constraints}) \\ & \mathbf{D}\mathbf{x} \geq \mathbf{d} \quad (\text{"easy" constraints}) \end{aligned}$$

$$\mathbf{x} \geq \mathbf{0}$$

In which, $\mathbf{Dx} \geq \mathbf{d}$ are the constraints we know how to deal with, and $\mathbf{Ax} \geq \mathbf{b}$ are the rest of constraints (that we do not like). Recall the Minkowski-Weyl Theorem again. For those "easy" constraints, i.e., the polyhedron $X = \{\mathbf{x} \geq \mathbf{0} | \mathbf{Dx} \geq \mathbf{d}\}$, can be represented by a set of extreme points $V = \{\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_{|V|}}\}$ and a set of extreme rays $R = \{\mathbf{r_1}, \mathbf{r_2}, \cdots, \mathbf{r_{|R|}}\}$. Replace the $\mathbf{Dx} \geq \mathbf{d}$ constraints with V and R, the origin (LP) model can be reformulated by substitution of $\mathbf{Dx} \geq \mathbf{d}$, as follows

$$\begin{aligned}
(LP) \quad \min \quad & \mathbf{c}^\top (\sum_{\mathbf{v} \in \mathbf{V}} \lambda_{\mathbf{v}} \mathbf{x_v} + \sum_{\mathbf{r} \in \mathbf{R}} \lambda_{\mathbf{r}} \mathbf{x_r}) \\
\text{s.t.} \quad & \mathbf{A}(\sum_{v \in V} \lambda_v \mathbf{x_v} + \sum_{r \in R} \lambda_r \mathbf{x_r}) \geq \mathbf{b} \\
& \sum_{v \in V} \lambda_v = 1 \\
& \lambda_v \geq 0 \quad \forall v \in V \\
& \lambda_r \geq 0 \quad \forall r \in R
\end{aligned}$$

For each x_v and x_r, define

$$\begin{aligned}
c_v &= \mathbf{c}^\top \mathbf{x_v}, \forall v \in V \\
c_r &= \mathbf{c}^\top \mathbf{x_r}, \forall r \in R \\
a_v &= \mathbf{A}\mathbf{x_v}, \forall v \in V \\
a_r &= \mathbf{A}\mathbf{x_r}, \forall r \in R
\end{aligned}$$

The (LP) model is reformulated as the master problem (MP)

$$\begin{aligned}
\text{(MP)} \quad z^*_{MP} = \min \quad & \sum_{v \in V} c_v \lambda_v + \sum_{r \in R} c_r \lambda_r \\
\text{s.t.} \quad & \sum_{v \in V} a_v \lambda_v + \sum_{r \in R} a_r \lambda_r \geq \mathbf{b} \quad [\pi] \\
& \sum_{v \in V} \lambda_v = 1 \quad [\pi_0] \\
& \lambda_v \geq 0 \quad \forall v \in V \\
& \lambda_r \geq 0 \quad \forall r \in R
\end{aligned}$$

There are two cases for reduce cost calculation:

- for $\lambda_v, v \in V$

$$\bar{c}_v = c_v - \begin{bmatrix} \pi^\top & \pi_0 \end{bmatrix} \begin{bmatrix} \mathbf{a_v} \\ 1 \end{bmatrix} = c_v - \pi^\top \mathbf{a_v} - \pi_0$$

- for $\lambda_r, r \in R$

$$\bar{c}_r = c_r - \begin{bmatrix} \pi^\top & \pi_0 \end{bmatrix} \begin{bmatrix} \mathbf{a_r} \\ 0 \end{bmatrix} = c_r - \pi^\top \mathbf{a}_r$$

The smallest reduce cost is derived by $\bar{c}^* = \min\{\min_{v \in V}\{\bar{c}_v\}, \min_{r \in R}\{\bar{c}_r\}\}$, which is the so called Dantzig-Wolfe pricing problem.

$$z^*_{PP} = \min_{j \in V \cup R} c_j - \pi^\top \mathbf{a}_j$$

In this formulation, both extreme points and extreme directions are considered. Since π_0 is a constant, we can ignore it as it is in the objective function.

Replace the decision variables back to the origin ones in the Dantzig-Wolfe pricing problem.

$$\begin{aligned} z^*_{PP} &= \min_{j \in V \cup R} c_j - \pi^\top \mathbf{a}_j \\ &= \min_{j \in V \cup R} \mathbf{c}^\top \mathbf{x}_j - \pi^\top \mathbf{A}\mathbf{x}_j \end{aligned}$$

which is

$$\begin{aligned} \min \quad & (\mathbf{c}^\top - \pi^\top \mathbf{A})\mathbf{x} \quad \text{(dual of "difficult" constraints)} \\ \text{s.t.} \quad & \mathbf{D}\mathbf{x} \geq \mathbf{d} \quad \text{("easy" constraints)} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Each time when a DW pricing problem is solved, one of the following three cases will happen

- $z^*_{PP} = -\infty$. An extreme ray is identified, add λ_{r^*} to the RMP with cost $\mathbf{c}^\top \mathbf{x}_{\mathbf{r}^*}$ and column coefficient $\begin{bmatrix} \mathbf{A}\mathbf{x}_{\mathbf{r}*} \\ 0 \end{bmatrix}$
- $-\infty < z^*_{PP} - \pi_0 < 0$. An extreme point is identified, add λ_{v^*} to the RMP with cost $\mathbf{c}^\top \mathbf{x}_{\mathbf{v}^*}$ and column coefficient $\begin{bmatrix} \mathbf{A}\mathbf{x}_{\mathbf{v}*} \\ 1 \end{bmatrix}$
- $0 \leq z^*_{PP} - \pi_0$. STOP, $V \cup R = \emptyset$

3 Cutting Stock Problem

There are a stock of rods with length W, we need a set of items, J, each item $j \in J$ has a length of w_j for b_j copies. The objective is to obtain the demanded number of copies of each item by cutting the minimum possible number of stocks. Assuming there are at most M rods, which is sufficient to satisfy all demands. Let x_{ij} be the number of item j cut from stock i, and y_i be binary variables that indicates where stock i is used. The cutting stock problem is then defined as follows

$$
\begin{aligned}
\min \quad & z = \sum_{i=1}^{M} y_i \\
\text{s.t.} \quad & \sum_{i=1}^{M} x_{ij} \geq b_j, \quad j \in J \\
& \sum_{j \in J} w_j x_{ij} \leq W y_i, \quad i \in \{1, 2, \cdots, M\} \\
& x_{ij} \in \mathbb{Z}_+^{M \times |J|}, \quad i \in \{1, 2, \cdots, M\}, j \in J \\
& y_i \in \{0, 1\}, \quad i \in \{1, 2, \cdots, M\}
\end{aligned}
$$

3.1 (Restricted) master problem

A cutting pattern is a possible way of cutting a stock, defined by the number of copies of each items obtained from that stock.

$$
\begin{aligned}
\min \quad & z = \sum_{q=1}^{Q} \lambda_q \\
\text{s.t.} \quad & \sum_{q=1}^{Q} p_q \lambda_q \geq b_j, \quad j \in J \quad (\pi) \\
& \lambda_q \in \mathbb{Z}, \quad q = 1, 2, \cdots, Q
\end{aligned}
$$

In which, there are exponential number of λ variables, one for each cutting pattern, from 1 to Q. Let p_{qj} indicate how many copies of item j are obtained in the qth cutting pattern.

3.2 Pricing subproblem

At each iteration, the following Integer Knapsack Problem is solved

$$
\begin{aligned}
\min \quad & \bar{c} = 1 - \sum_{j \in J} \pi_j x_j \\
\text{s.t.} \quad & \sum_{j \in J} w_j x_j \leq W \\
x_j \in \mathbb{Z}_+^{|J|}, \quad & \forall j \in J
\end{aligned}
$$

Each solution of this model is a cutting pattern.

4 Vehicle Routing Problem with Time Windows

Consider a fleet of vehicles V, a set of customers C, and a directed graph $G = (C \cup \{0\} \cup \{|C|+1\}, N)$. (0 as the starting depot and $|C| + 1$ as a duplicate of depot for vehicles to return to). The cost of each arc is c_{ij} and the travel time is t_{ij}. Each vehicle has a capacity of q and each customer i has a demand of d_i. Each customer is associated with a time window $[a_i, b_i]$.

Let x_{ijk} be a binary decision variable, where

$$x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ drives directly from vertex } i \text{ to } j \\ 0, & \text{Otherwise} \end{cases}$$

and the decision variable s_{ik} is defined for each vertex i and vehicle k, it denotes the time vehicle k starts to serve customer i. The MILP formulation for VRPTW is described as follows

$$\begin{aligned}
\min \quad & \sum_{k=1}^{|V|} \sum_{i=0}^{|C|} \sum_{j=1}^{|C|+1} c_{ij} x_{ijk} \\
\text{s.t.} \quad & \sum_{k=1}^{|V|} \sum_{j=1}^{|C|+1} x_{ijk} = 1, \quad \forall i \in C \quad \text{(Coupling constraints, no } k \in V) \\
& \sum_{i=1}^{|C|} d_i \sum_{j=1}^{|C|+1} x_{ijk} \leq q, \quad \forall k \in V \\
& \sum_{j=1}^{|C|+1} x_{0jk} = 1, \quad \forall k \in V \\
& \sum_{i=0}^{|C|} x_{ihk} = \sum_{j=1}^{|C|+1} x_{hjk} \quad \forall h \in C, k \in V \\
& \sum_{i=0}^{|C|} x_{i,|C|+1,k} = 1, \quad \forall k \in V \\
& s_{ik} + t_{ij} - M_{ij}(1 - x_{ijk}) \leq s_{jk}, \quad \forall i \in C \cup \{0\}, j \in C \cup \{|C|+1\}, k \in V \\
& a_i \leq s_{ik} \leq b_i, \quad \forall i \in C \cup \{0\} \cup \{|C|+1\}, k \in V \\
& x_{ijk} \in \{0,1\}, \quad \forall i \in C \cup \{0\} \cup \{|C|+1\}, k \in V
\end{aligned}$$

Now, the question is, who are the "difficult" constraints? Notice that there is only one set of coupling constraints ($\sum_{k=1}^{|V|} \sum_{j=1}^{|C|+1} x_{ijk} = 1$), and the remaining constraints are dealing with each vehicle separately. (They all have $\forall k \in V$), therefore, for sure, that coupling constraints should remain in the master problem. One of the approaches to is to keep the coupling constraints in the master problem.

4.1 (Restricted) Master Problem

The master problem is a set partitioning problem and the subproblem becomes a Shortest Path Problem with Time Windows and Capacity Constraints (SPPTWCC) or more specifically, the Elementary Shortest Path Problem with Time Windows and Capacity Constraints (ESPPTWCC).

The restricted master problem is defined as follows

$$\text{(RMP)} \quad \min \quad \sum_{r \in R'} c_r y_r$$

$$
\begin{aligned}
\text{s.t.} \quad & \sum_{r \in R'} a_{ir} y_r = 1, \quad \forall i \in C \quad [\pi] \\
& -\sum_{r \in R'} y_r \geq -K \quad [\pi_0] \\
& y_r \geq 0, \quad \forall r \in R' \quad \text{(exponential number of } r\text{s)}
\end{aligned}
$$

For each feasible route r (an extreme point), c_r is the length of route r, and a_{ir} is the 0-1 parameter that take 1 if vertex i is visited by route r. Let w_{ij} be binary variables which takes 1 if the sub-path go direct from vertex i to j. Then,

$$
\begin{aligned}
c_r &= \sum_{(i,j) \in E} c_{ij} w_{ij} \\
a_{ir} &= \sum_{(i,j) \in E} w_{ij}
\end{aligned}
$$

4.2 Pricing subproblem

The pricing subproblem is defined as follows,

$$
\begin{aligned}
\text{(PP)} \quad \min \quad & \sum_{(i,j) \in E} (c_{ij} - \pi_i) w_{ij} \\
\text{s.t.} \quad & \sum_{j:(i,j) \in E} w_{ij} - \sum_{j:(j,i) \in E} w_{ji} = 0, \quad \forall i \in C \\
& \sum_{j:(0,j) \in E} w_{0j} = 1 \\
& \sum_{j:(j,0) \in E} w_{j0} = 1 \\
& \sum_{(i,j) \in E} w_{ij} d_i \leq q \\
& t_j \geq t_i + d_{ij} - (1 - w_{ij}) M, \quad \forall i \in C, j \in C \\
& a_i \leq t_i \leq b_i, \quad \forall i \in C
\end{aligned}
$$

Notice that we would prefer to have the subproblem "easy to solve". Although the ESPPTWCC is strongly NP-hard, we can still use efficient (compare to IP) methods such as labeling algorithms to speed up the subproblem solving.

5 Early Branching v.s. Branch-and-Price

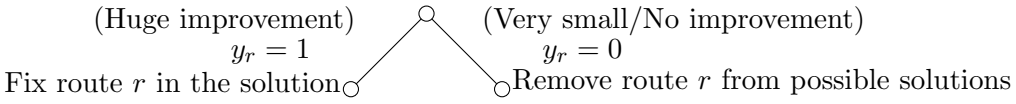
For MILP problems, the master problem in the Column Generation is an LP relaxation, thus it does not always produce integral solutions (if an integral solution is found, we reach the optimal solution). To derive the optimal solution, in general there are two approaches: early branching heuristic and the Branch-and-Price.

The early branching heuristic is a two-step approach. In the first step, do CG, until sufficient number of columns/routes are generated (with $y_r \geq 0$). Then, switch the definition of y_rs from $\mathbb{R}$ to

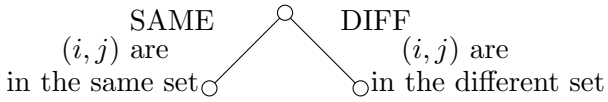
$\{0, 1\}$, solve the IP. It is called early branching because it branches "early" with partial information. The downsides are, first, the subset of the routes might not always create a feasible solution for VRPTW, and second, the heuristic solution might be far away from the optimal solution.

The other method is to incorporate the classic Branch-and-Bound strategy in the integer solution searching. The general idea is to replace the LP solving Simplex Methods in each iteration with a Column Generation subroutine.

There are several critical problems/challenges with Branch-and-Price for VRPTW (which is why my professor who taught me B&P don't like it, but somehow many researchers are fascinated with it... for some reasons). First, due to 0-1 branching, the B&P will create a **very** unbalance search tree. Take the following search tree as an example.



The right sub-tree will continue to be searched very inefficiently, due to the unbalance of the tree. The good news is, there are possible improvements to this issue by applying different branching strategies. One of them is the Ryan-Foster Branching Rule, which is designed for the Set Covering Problem. For any fractional solution, there are at least two elements (i, j) so that i and j are both partially covered by the same set S, but there is another set T that only covers i



For the search go through the SAME branch, the following constraint will be added into the follow-up subproblems.

$$\sum_{l:(i,l)\in E} w_{il} = \sum_{l:(j,l\in E)} w_{jl}$$

For the DIFF branch, the corresponded constraints will be

$$\sum_{l:(i,l)\in E} w_{il} + \sum_{l:(j,l\in E)} w_{jl} \leq 1$$

Second, there is a terrible symmetric issue/degeneracy issue in the subproblem - the subproblem will produce duplicate routes. One of the another solutions is for every route r in $y_r = 0$ branches, add the following constraints

$$\sum_{(i,j)\in E, w_{ij}^r = 0, \forall r\in R'} w_{ij} + \sum_{(i,j)\in E, w_{ij}^r = 1, \forall r\in R'} (1 - w_{ij}) \geq 1$$

The idea is to use the edges that have not yet been used more, and use the edges that have appeared in other edges less. (But it will still be bad...)