

Operations Research Operations Research Operations Research Operations Research
Operations Research Operations Research Operations Research Operations Research
Operations Research Operations Research Operations Research Operations Research
Operations Research Operations Research Operations Research Operations Research

Operations Research - Lecture 8

Olariu E. Florentin

November 24, 2025

1

Column Generation

- Introduction
- The algebra of Column Generation

2

Examples

- Cutting stock problem
- Constraint shortest path problem
- Vehicle routing problem

3

Dantzig-Wolfe decomposition principle in LP

4

Bibliography

- **Column generation** refers to algorithms for solving linear programming problems for which the number of variables is huge compared to the number of constraints.
- If the number of variables is exponentially in the size of the problem, then the ILP problem cannot be solved.
- Since most of the variables will be non-basic (and have a value of zero) with respect to an optimal basis, we can consider - at least theoretically - only a subset of variables, containing the basis.
- This is the crucial point: the number of non-zero variables (non-zero basic variables) is at most the number of constraints, hence even if the number of columns (the total number of variables) may be huge, we only need a small subset of all these variables for an optimal solution.

- This subset is composed only of the variables that have an estimated potential to improve the objective function.
- The original problem is split into: the (*restricted*) **master problem** and the **subproblem**.
- The *restricted master problem* keeps only a subset of variables being considered with the same (corresponding) objective function.
- The *subproblem* is a new problem created to identify a new (non-basic) variable which will become a basic variable.
- The objective function of the subproblem will be the reduced cost of the new variable with respect to the current dual variables.

In other words

- (i) *we start with a small number of variables (columns),*
- (ii) *solve the corresponding reduced LP problem,*
- (iii) *check if the solution can be improved by adding a new variable (column):*
 - if no, we can stop, an optimal solution was found;
 - if yes, add the column and go to the step (ii);
 - Adding a new variable means adding a new column, hence the name **Column Generation**.

The algebra of Column Generation

- Consider the following primal LP problem

$$\begin{aligned} & \text{minimize} && z = c^T x, \\ & \text{subject to} && Ax = b, \\ & && x \geq 0. \end{aligned} \tag{1}$$

- Its dual is

$$\begin{aligned} & \text{maximize} && w = b^T y, \\ & \text{subject to} && A^T y \leq c. \end{aligned} \tag{2}$$

- Suppose that $A \in \mathbb{R}^{m \times n}$ and its columns are A_1, A_2, \dots, A_n .
- We keep for the restricted master problem only a subset of variables:
 $I \subseteq [n] = \{1, 2, \dots, n\}$.

The algebra of Column Generation

- The primal problem can be written as

$$\begin{aligned} \text{minimize} \quad & z = \sum_{i=1}^n c_i x_i, \\ \text{subject to} \quad & \sum_{i=1}^n A_i x_i = b_i, \forall i \in [n] \\ & x_i \geq 0, \forall i \in [n]. \end{aligned} \tag{3}$$

- The *restricted master problem* is

$$\begin{aligned} \text{minimize} \quad & z = \sum_{i \in I} c_i x_i, \\ \text{subject to} \quad & \sum_{i \in I} A_i x_i = b_i, \forall i \in I \\ & x_i \geq 0, \forall i \in I. \end{aligned} \tag{4}$$

The algebra of Column Generation

- Let x be a basic feasible solution to (4); since the non-basic variables are all null, x will be a basic feasible solution to (3) also.
- We can order the variables such that $x^T = (x_B^T \ x_N^T)$, where x_B is the vector of basic variables and x_N is the vector of non-basic ones ($x_N = 0$); therefore $B \subseteq I$.

- Correspondingly, we split c and A :

$$c^T = (c_B^T \ c_N^T), \quad A = (B \ N).$$

- The objective function and the constraints are

$$z = c_B^T x_B + c_N^T x_N, \quad Bx_B + Nx_N = b.$$

- The basis is $x_B = B^{-1}b - B^{-1}Nx_N$, hence the objective function is

$$z = c_B^T B^{-1}b + (c_N^T - c_B^T B^{-1}N)x_N.$$

The algebra of Column Generation

- The reduced cost of a non-basic variable x_j is
$$c_j - c_B^T B^{-1} A_j.$$
- Now, $y = (c_B^T B^{-1})^T$ is the the vector of *simplex (or dual) multipliers* and also the solution to the dual problem of (4).
- Hence finding a variable to enter the basis (which is the first applied criteria in simplex algorithm methodology) means to find $j \in N$ such that
$$c_j - y^T A_j < 0.$$
- The *subproblem* is
$$j_0 = \arg \min_{j \in N} (c_j - y^T A_j).$$
- That is, we find the non-basic variable with the minimum reduced cost (the Dantzig rule).

The algebra of Column Generation

- This operation is called *pricing* a non-basic variable to enter the basis.
- After the variable is chosen its column is added to the current *restricted master problem* and the procedure reiterates: I becomes $I \cup \{j_0\}$, the restricted master problem is optimized again, another non-basic variable is priced out, ...
- If the minimum in the subproblem is non-negative, the optimal solution of the restricted master problem is optimal to the original problem too. This is the stopping criteria for the above loop.

The algebra of Column Generation

Remarks:

- After solving the *subproblem* one can add more than one variable to the *restricted master problem*.
- But adding too much variables may result in a very large *restricted master problem* which will need more resources (CPU time and memory) to solve.
- In an intermediate step is not necessarily to add variables for which the reduced cost is minimum (and negative), it is enough that this reduced cost to be negative. But usually such a strategy may require a larger number of subproblems to be solved - this is an empirical observation.
- **Column Generation is prone to numerical errors.**

The algebra of Column Generation

Final remarks:

- Sometimes an heuristic for finding an initial set of variables is used before employing the *Column Generation*.
- The *Column Generation* approach depends on the type of the *sub-problem*: if this problem cannot be (easily) solved, then the method is useless.
- If *Column Generation* is combined with *Branch & Bound*, the result is an algorithm called *Branch & Price*.

Cutting stock problem

- The cutting stock problem arises in various industrial applications.
- We have an unlimited number of stocks of width L and need to cut n_i stocks of width l_i , for any $i \in [m]$.
- The goal is to minimize the number of stocks used.
- We must use the set of cutting patterns \mathcal{C} - which usually is huge in size.
- Denote by a_{ic} the number of items of stock i we get from the pattern $c \in \mathcal{C}$. The following inequalities must hold

$$\sum_{i=1}^m a_{ic} l_i \leq L, \forall c \in \mathcal{C}.$$

- Denote by x_c the number of times the cutting pattern c is used.

- The cutting stock problem as an ILP problem is:

$$\begin{aligned} & \text{minimize} && z = \sum_{c \in C} x_c, \\ & \text{subject to} && \sum_{c \in C} a_{ic} x_c \geq n_i, \forall i \in [m] \\ & && x_c \in \mathbb{Z}_+, \forall c \in C. \end{aligned} \quad (5)$$

- We first solve the relaxed problem.
- Consider a small subset of patterns, C' , of cardinality at least m , and build the *restricted master problem*.

$$\begin{aligned} & \text{minimize} && z = \sum_{c \in C'} x_c, \\ & \text{subject to} && \sum_{c \in C'} a_{ic} x_c \geq n_i, \forall i \in [m] \\ & && x_c \geq 0, \forall c \in C'. \end{aligned} \quad (6)$$

Cutting stock problem

- Let $y = (y_1, y_2, \dots, y_m)$ be a solution to the dual of (6).
- The *subproblem* is
$$c_0 = \arg \min_{c \in \mathcal{C}} \left(1 - \sum_{i=1}^m y_i a_{ic} \right).$$
- Finding a pattern for cutting a stock means to find $(a_{ic})_{i \in [m]}$ such that
$$\sum_{i=1}^m a_{ic} l_i \leq L, \text{ with } a_{ic} \in \mathbb{Z}_+, \forall i \in [m].$$
- Since the set of all patterns is not known, we can drop the index $c \in \mathcal{C}$.

- Therefore we have to solve the following new ILP (sub)problem:

$$\begin{aligned} & \text{maximize} && z = \sum_{i=1}^m y_i a_i, \\ & \text{subject to} && \sum_{i=1} a_i l_i \leq L \\ & && a_i \in \mathbb{Z}_+. \end{aligned} \tag{7}$$

- If the minimum in the above subproblem is at most 1, then the optimal solution to restricted master problem is optimal to the original problem too.
- This is the classical *knapsack* problem which is an NP-hard problem but can be solved using, for example, dynamic programming even for large values of m .

Cutting stock problem

- The column generation method is a method for solving the relaxation of (6).
- After that we can find an integer solution (optimal or not) for the original problem
 - ▶ by rounding up an optimal solution of the relaxed or
 - ▶ by using an heuristic or
 - ▶ using branch&bound (which becomes branch & price), branch & cut etc.

Constraint shortest path problem

- Suppose we have a digraph with costs on its arcs, but also a resource consumption (like the traversal time). s and t are two given vertices in this digraph.
- The **Constraint shortest path problem** is an optimization problem that requires to find a minimum cost st -path whose total traversal time is upper bounded by a constant T .
- The corresponding decision problem is

Instance: $G = (V, E)$ a digraph, $c: E \rightarrow \mathbb{Z}_+$, $t: E \rightarrow \mathbb{R}_+$, $C, T \in \mathbb{R}_+$, and $s, t \in V$.

Question: There is a st -path of cost at most C , whose total traversal time is at most T ?

Constraint shortest path problem

- The ILP formulation of constraint shortest path problem is

$$\begin{aligned} & \text{minimize} && \sum_{ij \in E} c_{ij} x_{ij}, \\ & \text{subject to} && \sum_{sj \in E} x_{sj} = 1, \\ & && \sum_{jt \in E} x_{jt} = 1, \\ & && \sum_{j:ij \in E} x_{ij} = \sum_{j:ji \in E} x_{ji}, \quad \forall i \in V \setminus \{s, t\} \\ & && \sum_{ij \in E} t_{ij} x_{ij} \leq T, \\ & && x_{ij} \in \{0, 1\}. \end{aligned} \tag{8}$$

- Even is an *almost* shortest path problem, constraint shortest path problem is NP-complete (even for $C, T \in \mathbb{Z}_+$).

Constraint shortest path problem

- If we ignore the resource limitation constraint the remaining constraints will give a minimum cost flow problem.
- Any flow in a transportation network corresponding to an extreme point of the (relaxed) polytope is, in fact, a *st*-path.
- Therefore a flow will be a convex combination of paths from the family, \mathcal{P} , of all *st*-paths in G .
- We get in this way a new form of the above LP problem, by replacing each arc-flow variable by the corresponding convex combination of path flows:

$$\begin{aligned}x_{ij} &= \sum_{P \in \mathcal{P}} \alpha_p x_{ij}^P, \forall ij \in E, \\ \sum_{P \in \mathcal{P}} \alpha_p &= 1, \\ \alpha_p &\geq 0, \forall P \in \mathcal{P}.\end{aligned}\tag{9}$$

Constraint shortest path problem

- The new problem is

$$\begin{aligned} & \text{minimize} && \sum_{P \in \mathcal{P}} \alpha_P \left(\sum_{ij \in E} c_{ij} x_{ij}^P \right), \\ & \text{subject to} && \sum_{P \in \mathcal{P}} \alpha_P \left(\sum_{ij \in E} t_{ij} x_{ij}^P \right) \leq T, \\ & && \sum_{P \in \mathcal{P}} \alpha_P = 1, \\ & && \sum_{P \in \mathcal{P}} \alpha_P x_{ij}^P = x_{ij}, \quad \forall ij \in E, \\ & && \alpha_P \geq 0, \quad \forall P \in \mathcal{P}, \\ & && x_{ij} \in \{0, 1\}, \end{aligned} \tag{10}$$

where $x_{ij}^P = 1$ if and only if $ij \in E(P)$.

- (By keeping the arc-flow variables we can recover a solution to the original problem.)

Constraint shortest path problem

- We start by relaxing the problem (10), i. e., we replace the integrality constraints for the arc-flow variables with

$$x_{ij} \geq 0, \forall ij \in E.$$

- In this way there is no need of arc-flow variables, hence the remaining problem has only two constraints:

$$\begin{aligned} & \text{minimize} && \sum_{P \in \mathcal{P}} \alpha_P \left(\sum_{ij \in E} c_{ij} x_{ij}^P \right), \\ & \text{subject to} && \sum_{P \in \mathcal{P}} \alpha_P \left(\sum_{ij \in E} t_{ij} x_{ij}^P \right) \leq T, \\ & && \sum_{P \in \mathcal{P}} \alpha_P = 1, \\ & && \alpha_P \geq 0, \quad \forall P \in \mathcal{P}. \end{aligned} \tag{11}$$

Constraint shortest path problem

- The cardinality of \mathcal{P} is very large, sometimes huge, and we cannot enumerate all the st -paths in G .
- Hence we will start with a small subset of such paths (variables) ensuring that the restricted master problem is feasible; more variables will be added only when needed.
- Let y_0 and y_1 be the dual (multipliers) variables for problem (11).
- After solving the restricted master problem we try to find a new variable α_P (that is, a new st -path) having a negative reduced cost.
- This is the subproblem:

$$P_0 = \arg \min_{P \in \mathcal{P}} (c(P) - y^T A_P). \quad (12)$$

Constraint shortest path problem

- Equivalently, if (y_0, y_1) is an optimal dual solution,

$$P_0 = \arg \min_{P \in \mathcal{P}} \left[\sum_{ij \in E} c_{ij} x_{ij}^P - y_0 \left(\sum_{ij \in E} t_{ij} x_{ij}^P \right) - y_1 \right]. \quad (13)$$

- If the optimum in (13) is negative the variable and the column corresponding to P_0 will be added to the restricted master problem.
- (13) gives a subproblem which requires to find a minimum cost path (provided that this cost is $< y_1$).
- Otherwise the current optimal solution to the restricted master problem is an optimal solution to the original (master) relaxed problem.
- Because the original problem was an ILP we can embed this method within a *Branch & Price* algorithm.
- In each node of the *B & P* tree we will use again the *Column generation method* for solving the relaxed current problem.

Vehicle routing problem

- The goal is to find optimal routes for multiple vehicles visiting a set of customers.
- When there's only one vehicle, it reduces to the Traveling Salesman Problem. Hence this problem is NP-hard also.
- The basic model of vehicle routing problem:
 - ▶ a number of vehicles located at a central depot (s) has to deliver a commodity to a set of n customers which surround the depot.
 - ▶ each vehicle has a given capacity (C) and each customer i has a given demand (d_i).
 - ▶ we want to minimize the total travel distance.
- We will describe two ILP models for this problem.

Vehicle routing problem - first model

- Suppose that $G = (V, E)$ is the digraph and $c : E \rightarrow \mathbb{R}_+$ is the transportation cost.

$$\text{minimize} \quad \sum_{ij \in E} c_{ij} x_{ij},$$

$$\text{subject to} \quad \sum_{j \in V} x_{ij} = 1, \quad \forall i \in V \setminus \{s\},$$

$$\sum_{i \in V} x_{ij} = \sum_{k \in V} x_{jk}, \quad \forall j \in V,$$

$$C(1 - x_{ij}) + y_i \geq y_j + d_j, \quad \forall i \in V, j \in V \setminus \{s\},$$

$$y_i \leq C, \quad i \in V$$

$$x_{ij} \in \{0, 1\}.$$

(14)

- y_i represent the load of the vehicle when leaving vertex i .

Vehicle routing problem - second model

- Suppose we can generate the set of all possible routes, denoted by \mathcal{R} .
- That is, \mathcal{R} is the set of all tours starting from the depot, visiting some customers exactly once and coming back to the depot.
- In other words a feasible route is a cycle going through the depot vertex s .
- A solution to the vehicle routing problem is a collection of disjoint feasible routes covering all customers.
- Let c_R be the (transportation) cost of route $R \in \mathcal{R}$ and $b_{iR} = 1$ if customer i is served along the route R , and $b_{iR} = 0$ otherwise.

Vehicle routing problem - second model

- The second model is

$$\begin{aligned} & \text{minimize} && \sum_{R \in \mathcal{R}} c_R x_R, \\ & \text{subject to} && \sum_{R \in \mathcal{R}} b_{iR} x_R \geq 1, \quad \forall i \in V \setminus \{s\}, \\ & && x_R \in \{0, 1\}, \quad \forall R \in \mathcal{R}. \end{aligned} \quad (15)$$

- Usually the number of routes is huge and we cannot list them all (together with their costs and coefficients b_{iR}).
- We can use the *Column generation* method as long as we have a method for generating feasible routes.

- Start with a few routes that make the problem feasible and suppose that after adding some variables the set of routes is R' ; define the restricted master problem, its relaxation is

$$\begin{aligned} & \text{minimize} && \sum_{R \in \mathcal{R}'} c_R x_R, \\ & \text{subject to} && \sum_{R \in \mathcal{R}'} b_{iR} x_R \geq 1, \quad \forall i \in V \setminus \{s\}, \\ & && x_R \geq 0, \quad \forall R \in \mathcal{R}'. \end{aligned} \quad (16)$$

- We consider the dual of problem (16) and $y = (y_i)_{i \in V \setminus \{s\}}$ an optimal solution of it:

$$\begin{aligned} & \text{maximize} && \sum_{i \in V \setminus \{s\}} y_i, \\ & \text{subject to} && \sum_{i \in V \setminus \{s\}} b_{iR} y_i \leq c_R, \quad \forall R \in \mathcal{R}', \\ & && y_i \geq 0, \quad \forall i \in V \setminus \{s\}. \end{aligned} \quad (17)$$

Vehicle routing problem

- We then search for a route $R \in \mathcal{R}$ (a column) with negative reduced cost:

$$c'_R = c_R - \sum_{i \in V \setminus \{s\}} b_i R y_i < 0.$$

- The subproblem: find a route with minimum reduced cost.
- $x_{ij} = 1$ if the arc ij is on the route and $x_{ij} = 0$ otherwise, then

$$c_R = \sum_{ij \in E} c_{ij} x_{ij}$$

- Therefore, the reduced cost will become

$$c'_R = c_R - \sum_{i \in V \setminus \{s\}} b_i R y_i = \sum_{ij \in E} c'_{ij} x_{ij},$$

where $c'_{ij} = c_{ij} - y_i$.

Vehicle routing problem

- The subproblem requires to find a cycle through s of minimum cost, provided that this minimum is negative.
- One can use the Floyd-Warshall algorithm for finding a negative cost cycle containing s .
- An alternative is to solve the following ILP problem:

$$\begin{aligned} & \text{minimize} && \sum_{ij \in E} (c_{ij} - y_i) x_{ij}, \\ & \text{subject to} && \sum_{j \in V \setminus \{s\}} x_{sj} = 1, \\ & && \sum_{i \in V} x_{ij} = \sum_{k \in V} x_{jk}, \quad \forall j \in V, \\ & && x_{ij} \in \{0, 1\} \end{aligned} \tag{18}$$

and keep just the cycle through s .

- Consider the following LP problem - the *compact formulation*

$$\begin{aligned}
 & \text{minimize} && z = c^T x, \\
 & \text{subject to} && Ax \geq b, \\
 & && Dx = d, \\
 & && x \in \mathbb{Z}_+^n.
 \end{aligned} \tag{19}$$

- Suppose that the polyhedron $\mathcal{P} = \{x \in \mathbb{R}_+^n : Dx = d\}$ is not empty.
- From the representation theorem (see Lecture 1), each $x \in \mathcal{P}$ can be written as

$$x = y + \sum_{i=1}^t \lambda_i v^i, \tag{20}$$

where $Ay = 0$ (y is zero or is a direction of unboundedness), v^i are the extremal points for \mathcal{P} , and $\sum_{i=1}^t \lambda_i = 1, \lambda_i \geq 0, \forall i = \overline{1, t}$.

- Substituting for x in (19) and applying the linear transformations $\gamma_i = c^T v_i$, $\gamma = c^T y$, and $a_i = A v_i$, we get the *extensive formulation*:

$$\begin{aligned}
 & \text{minimize} && z = \sum_{i=1}^t \gamma_i \lambda_i + \gamma, \\
 & \text{subject to} && \sum_{i=1}^t a_i \lambda_i \geq b, \\
 & && \sum_{i=1}^t \lambda_i = 1, \\
 & && x = y + \sum_{i=1}^t \lambda_i v_i, \\
 & && \lambda_i \geq 0, 1 \leq i \leq t, \\
 & && x \in \mathbb{Z}_+^n.
 \end{aligned} \tag{21}$$

- After relaxing the integrality constraints of x , there is no need to link x and λ_i .
- The relaxation of (19) is

$$\begin{aligned} & \text{minimize} && z = \sum_{i=1}^t \gamma_i \lambda_i + \gamma, \\ & \text{subject to} && \sum_{i=1}^t a_i \lambda_i \geq b, \\ & && \sum_{i=1}^t \lambda_i = 1, \\ & && \lambda_i \geq 0, 1 \leq i \leq t. \end{aligned} \quad (22)$$

- $\sum_{i=1}^t \lambda_i = 1$ is called the *convexity constraint*.

- We can solve this master problem, (22), by column generation; the columns corresponds to the extreme points of \mathcal{P} (and possible to a direction of unboundedness).
- The initial restricted master problem will have a number of columns corresponding to (usually at least $n + 1$) extreme points of \mathcal{P} and maybe a direction of unboundedness.
- Let $\pi \in \mathbb{R}^t$ and π_0 be an optimal solution to the dual of the restricted master problem, where π_0 corresponds to the convexity constraint.
- Keeping the direction of unboundedness outside the first restricted master problem, we get a new form of the subproblem.

- The subproblem is

$$i_0 = \arg \min_{1 \leq i \leq t} (c_i - \pi^T a_i - \pi_0).$$

- By the above linear transformations the subproblem becomes

$$\begin{aligned} & \text{minimize} && c^* = (c^T - \pi^T A)x - \pi_0, \\ & \text{subject to} && Dx \geq d, \\ & && x \geq 0. \end{aligned} \quad (23)$$

- When $c^* \geq 0$, no negative reduced cost column exists, and the column generation algorithm ends.
- If $c^* < 0$ is finite, the optimal solution in (23) is an extremal point v_i of \mathcal{P} , and we can add to the restricted master problem the column $((Av_i)^T, 1, c^T v_i)^T$.




Dantzig-Wolfe decomposition

- If $c^* = -\infty$, we identify a direction of unboundedness y of \mathcal{P} , and we can add to the restricted master problem the column $(0^T, 0, c^T v_i)^T$.
- Dantzig-Wolfe type approximation algorithms with guaranteed convergence rates have been proposed for certain linear programs.
- For practical problems, usually this type of decomposition exploits a block diagonal structure of matrix D .
- It can be observed that the initial problem (19) was reduced to a problem with a smaller number of constraints on the expense of the number of variables (and the number of extremal points could be exponentially).

Dantzig-Wolfe decomposition

- The *Dantzig-Wolfe* decomposition is a way that theoretically reduces the size of a problem, since the column generation method works with problems having a very large number of variables.
- Another type of decomposition is *Benders'* which exploit also the block structure of the system matrix, but it uses instead of column generation a technique of adding constraints (*Benders cuts*) after trying to solve the first stage problem.

Bibliography

-  Dantzig, G. B., P. Wolfe, *Decomposition principle for linear programs*, Operations Research, 8, pp. 101-111, 1960.
-  Desrosiers, J., M. E. Lubbecke, *A primer in column generation in Column Generation*, Desaulniers, G., J. Desrosiers, M.M. Solomon (Eds.), Springer, 2005.
-  Desrosiers, J., M. E. Lubbecke, *Selected Topics in Column Generation*, Technical Report, Technische Universitat Berlin, 2004/08, 2004.