Notes on Dantzig-Wolfe decomposition and column generation

Mette Gamst

November 11, 2010

1 Introduction

This note introduces an exact solution method for mathematical programming problems. The method is based on Dantzig-Wolfe decomposition and delayed column generation. The note concerns mathematical programming where a problem is represented by an objective function and a set of constraints, e.g.:

$$\begin{aligned} \min \quad & cX \\ \text{s. t.} \quad & AX \geq b \\ & X \in S \end{aligned}$$

Here X is a vector of variables, all lying in the set S. The cost vector is denoted c. A is a matrix, b a vector, and together they form the constraints on the variables. When S is the set of real numbers, the mathematical model is said to be a linear program (LP). If S is a binary set or the set of integers, the problem is an integer program (IP). If some variables in X belong to the set of real numbers and others to the set of binaries or integers, the model is said to be a mixed integer program (MIP).

Let $X \in \mathbb{R} : \; x \geq 0, \; x \in X$ in the above formulation and denote this the *primal* problem. The *dual* model is found by transposing the primal problem:

$$\begin{aligned} \max \quad & b^T Y \\ \text{s. t.} \quad & A^T Y \leq c^T \\ & Y \geq 0 \end{aligned}$$

where b^T is the transpose of b, A^T is the transpose of A, and c^T is the transpose of c. The *dual variables* are denoted Y. The optimal solution value for the dual problem equals the optimal solution value for the primal problem.

Dantzig-Wolfe decomposition transforms the original mathematical problem into a *master problem*, where the number of columns may be large but the number of rows is reduced. To make the new model more tractable, columns are generated iteratively in the hopes of only having to include a subset of the columns in the model. This is denoted delayed column generation and consists of solving a *pricing problem* in each iteration.

This note introduces Dantzig-Wolfe decomposition and delayed columns generation and is not meant to be an in-depth survey but more a guide for understanding the basics of the approaches. For details on the methods and examples of applications, see e.g. Desaulniers et al. [3], Lübbecke and Desrosiers [4], and Nemhauser and Wolsey [5].

2 Dantzig-Wolfe decomposition

Dantzig-Wolfe decomposition was introduced by Dantzig and Wolfe [2] and consists of reformulating a problem into a master problem and a pricing problem for improving the tractability of large-scale problems. The master problem typically has fewer constraints than the original problem, but the number of columns may be very large. The pricing problem generates columns, which have the potential to improve the current solution.

In order to Dantzig-Wolfe decompose a problem, the constraint matrix should take on a certain structure and consist of a number of *independent* constraints and a number of *connecting* constraints. The constraint matrix is block-angular, i.e., the matrix can be divided into blocks with non-zero coefficients. These blocks constitute the independent constraints. Connecting constraints bind the columns together. Consider the problem:

$$\begin{aligned}
\min \quad & \sum_{k \in K} c^k x^k && (1)\\
\text{s. t.} \quad & \sum_{k \in K} A^k x^k \leq b && (2)\\
& D^k x^k \leq d^k \qquad \forall k \in K && (3)\\
& x^k \in \mathbb{Z}_+^{nk} \qquad \forall k \in K && (4)
\end{aligned}$$

where K is the set of blocks and A^k and D^k constitute the constraint matrices. Constraints A^k are the connecting block, and D^k the independent block.

Figure 1 illustrates this matrix consisting of connecting and independent constraints as blocks A^k and D^k, respectively. Now, we define the domains

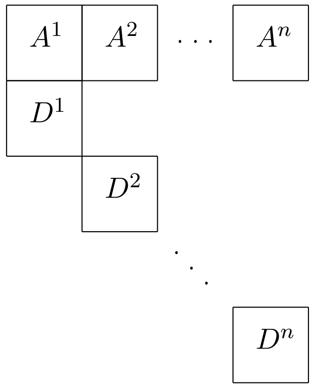

Figure 1: The desired matrix structure for Dantzig-Wolfe decomposition. The blocks $A^1,\ A^2,\ \ldots,\ A^n$ are connecting constraints and the blocks $D^1,\ D^2,\ \ldots,\ D^n$ are independent constraints.

X^k as $X^k = \{x^k \in \mathbb{Z}_+^{nk}, D^k x^k \leq d^k\}$ and we can rewrite our problem into:

$$\min \quad \sum_{k \in K} c^k x^k \tag{5}$$

$$\text{s. t.} \quad \sum_{k \in K} A^k x^k \leq b \tag{6}$$

$$x^k \in X^k \qquad \forall k \in K \tag{7}$$

Note that this problem only contains the connecting constraints. The variables x^k must satisfy the independent constraints, which thus are left out. The model holds fewer constraints than the original formulation, but the number of columns may be very large. How to deal with the large number of variables is discussed in the next section.

Example: Consider the Minimum Cost Multi-Commodity unsplittable Flow Problem (MCMCuFP), which consists of sending a number of commodities through a capacitated network such that the total routing cost is minimized and such that each commodity uses exactly one path.

The network is represented as a graph with nodes and edges $G = (V, E)$. Commodities are represented by the set L and each commodity $l \in L$ consists of a source node, a target node, and a quantity q^l to route. Let $c_{ij} \geq 0$ be the cost of routing one unit of flow on edge $(ij) \in E$ and let d_{ij} be the capacity of edge $(ij) \in E$. Finally, let $x^l_{ij} \in \{0, 1\}$ be a binary variable indicating whether or not commodity

$l \in L$ visits edge $(ij) \in E$. Now MCMCuFP can be formulated as:

$$\min \quad \sum_{l \in L} \sum_{(ij) \in E} c_{ij} q^l x_{ij}^l \tag{8}$$

$$\text{s. t.} \quad \sum_{l \in L} q^l x_{ij}^l \leq d_{ij} \qquad \forall (ij) \in E \tag{9}$$

$$\sum_{(ij) \in E} x_{ij}^l - \sum_{(ji) \in E} x_{ji}^l = b_i^l \quad \forall i \in V,\ \forall l \in L \tag{10}$$

$$x_{ij}^l \in \{0, 1\} \qquad \forall (ij) \in E, \forall l \in L \tag{11}$$

The objective (8) minimizes the total cost of routing all commodities. The first constraint (9) ensures that edge capacities are not violated. In constraint (10) let $b_i^l = 1$ if i is the source node of commodity l, let $b_i^l = -1$ if i is the target node of commodity l, and let $b_i^l = 0$ otherwise. Constraint (10) ensures that each commodity is routed from its source node to its target node. Finally the bound (11) makes sure that variables take on binary values.

Barnhart et al. [1] Dantzig-Wolfe decomposed MCMCuFP such that the pricing problem generates a path for each commodity and the master problem merges the paths into an overall feasible solution. Let P be the set of paths and let the cost c_p of each path be defined as the sum of visited edges $\sum_{(ij) \in p} c_{ij}$. The binary variable $x_p^l \in \{0, 1\}$ indicates whether or not commodity $l \in L$ uses path $p \in P$. Also, let δ_{ij}^p be a constant denoting whether or not path p visits edge $(ij) \in E$. The master problem is:

$$\min \quad \sum_{l \in L} \sum_{p \in P} c_p q^l x_p^l \tag{12}$$

$$\text{s. t.} \quad \sum_{l \in L} q^l \delta_{ij}^p x_p^l \leq d_{ij} \quad \forall (ij) \in E \tag{13}$$

$$\sum_{p \in P} x_p^l = 1 \qquad \forall l \in L \tag{14}$$

$$x_p^l \in \{0, 1\} \qquad \forall p \in P, \forall l \in L \tag{15}$$

The objective (12) still minimizes the total cost of routing the commodities and the first constraint (13) makes sure that edge capacities are satisfied. Constraint (14) says that each commodity can use exactly one path and the bound (15) ensures that variables take on feasible values.

3 Delayed column generation

When applying Dantzig-Wolfe decomposition on an mathematical formulation, the number of columns may be very large. An idea is thus to only include a subset of the columns. In this case we denote (5)-(7) the *restricted master problem*, because only a subset of columns are included. Columns are generated iteratively by solving the pricing problem. Only columns, which have the potential to improve the current solution to the restricted master problem, are added. This procedure is denoted *delayed column generation*, or simply *column generation*.

To decide whether or not a column has potential to improve the current solution to the restricted master problem, the dual variables of the current solution are considered. Consider the restricted master problem:

$$\begin{aligned} \min \quad & \sum_{j \in J} c_j x_j \\ \text{s. t.} \quad & \sum_{j \in J} a_j x_j \geq b \\ & x_j \in X \end{aligned} \tag{16}$$

The *reduced cost* for a column $j \in J$ is defined as $c_j - ya_j$ where y is the dual cost vector. In minimization problems, a generated column has potential to improve the current solution to the restricted master problem if its reduced cost is negative; in maximization problems positive reduced costs are sought. Now, the objective of the pricing problem is the reduced cost and the constraints are the independent constraints of the original problem:

$$\begin{aligned} \min \quad & (c_j - ya_j) x_j \\ \text{s. t.} \quad & D x_j \leq d \\ & x_j \in \mathbb{Z}_+^n \end{aligned} \tag{17}$$

A pricing problem is generated for each block $k \in K$ of the original problem. The pricing problems for different blocks may thus differ. Columns generated by the pricing problem, are not necessarily part of the solution in the following iteration even though they had negative reduced costs. If one generated column becomes part of the next solution then the remaining generated columns may become uninteresting. Also, even if a column is part of the solution in the iteration just after its generation, the column is not necessarily part of an optimal solution.

The overall column generation procedure can now be stated as:

1. Solve the restricted master problem (16)

2. Generate columns with the most negative reduced costs by solving the corresponding pricing problems (17)

3. If new columns are generated go to step 1, otherwise stop

Often it is only slightly more expensive to generate several columns at a time. Hence this may be beneficial, for instance when the pricing problem is difficult to solve, e.g. $\mathcal{NP}$-hard. In this case, the pricing problem can also be solved heuristically. However, when the heuristic cannot generate a column with negative reduced cost, then the pricing problem must be solved to optimality to ensure that the column generation procedure eventually gives an optimal solution.

Example (cont). Consider the Minimum Cost Multi-Commodity unsplittable Flow Problem from the previous example and how the problem was Dantzig-Wolfe decomposed. This example shows how to generate columns for the master problem according to Barnhart et al. [1]. The restricted master problem became:

$$\min \quad \sum_{l \in L} \sum_{p \in P} c_p q^l x_p^l \tag{18}$$

$$\text{s. t.} \quad \sum_{l \in L} q^l \delta_{ij}^p x_p^l \leq d_{ij} \quad \forall (ij) \in E \tag{19}$$

$$\sum_{p \in P} x_p^l = 1 \qquad \forall l \in L \tag{20}$$

$$x_p^l \in \{0, 1\} \qquad \forall p \in P, \forall l \in L \tag{21}$$

Let $\pi_{ij} \leq 0$ be the dual of constraint (19) and $\sigma^l \in \mathbb{R}$ be the dual of constraint (20). The reduced cost for a column p for a commodity l is:

$$\bar{c}_p^l = \sum_{(ij) \in E} q^l (c_{ij} - \pi_{ij}) - \sigma^l$$

The pricing problem for each column p and commodity l seeks to find columns with negative reduced cost. Now, σ^l is known for each commodity and the reduced cost can be rewritten as:

$$\sum_{(ij) \in E} q^l (c_{ij} - \pi_{ij}) < \sigma^l$$

Let the cost of each edge $(ij) \in E$ in the graph be replaced by $(c_{ij} - \pi_{ij})$, which is non-negative because $c_{ij} \geq 0$ and $\pi_{ij} \leq 0$. The pricing problem consists of finding the shortest path from the source node to the target node of the commodity, such that the total (reduced) cost is minimized. Because edge weights are non-negative, the pricing problem is polynomially solvable. If the pricing problem finds a path with total cost less than σ^l then the corresponding column is priced into the master problem.

References

[1] C. Barnhart, C. A. Hane, and P. H. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2):318–326, 2000.

[2] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.

[3] G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors. *Column Generation.* Springer Science+Media Business, Inc., 2005.

[4] M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[5] G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization.* John Wiley & Sons, Inc., 1999.