Methods and Models for Combinatorial Optimization

Column generation methods

L. De Giovanni   M. Di Summa   G. Zambelli

1 A cutting stock problem

A company has a stock of iron rods with diameter 15 millimeters and length 11 meters and cuts the rods for its customers, who require different lengths. At the moment, the following demand has to be satisfied:

item type	length (m)	number of pieces required
1	2.0	48
2	4.5	35
3	5.0	24
4	5.5	10
5	7.5	8

Determine the minimum number of iron rods that should be used to satisfy the total demand.

A possible model for the problem, proposed by Gilmore and Gomory in 1960[1] is the following.

Sets

- $I = \{1, 2, 3, 4, 5\}$: set of item types;
- J: set of patterns (i.e., possible ways) that can be adopted to cut a single rod into pieces of the required lengths.

Parameters

- W: rod length (before the cutting);

[1]P.C.Gilmore and R.E.Gomory, "A linear programming approach to the cutting stock problem", Operations Research, Vol. 9, No. 6 (Nov.-Dec., 1961), pp. 849–859

- L_i: length of item $i \in I$;
- R_i: number of pieces of type $i \in I$ required;
- N_{ij}: number of pieces of type $i \in I$ in pattern $j \in J$.

Decision variables

- x_j: number of rods that should be cut using pattern $j \in J$.

Model

$$\begin{array}{rrcll} \min & \sum_{j \in J} x_j & & & \\ s.t. & \sum_{j \in J} N_{ij} x_j & \geq & R_i & \forall \ i \in I \\ & x_j & \in & \mathbb{Z}_+ & \forall \ j \in J \end{array}$$

2 Problem solution

The model is very elegant, but assumes the availability of the set J and the parameters N_{ij}. In order to generate this data, one needs to enumerate all possible cutting patterns. It is easy to realize that the number of possible cutting patterns is huge, and therefore a direct implementation of the above model is unpractical for real-world instances.

We remark that it makes sense to solve the continuous relaxation of the above model. This is because, in practical situations, the demands are so high that the number of rods cut is also very large, and therefore a good heuristic solution can be determined by rounding up to the next integer each variable x_j found by solving the continuous relaxation. Moreover, the solution of the continuous relaxation may constitute the starting point for the application of an exact solution method (for instance, the Branch-and-Bound).

> **We therefore analyze how to solve the *continuous relaxation* of the model.**

As a starting point, we need a feasible solution for the linear relaxation. Such a solution can be constructed as follows:

1. consider single-item cutting patterns, i.e., $|I|$ configurations, each containing $N_{ii} = \lfloor W/L_i \rfloor$ pieces of type i;
2. set $x_i = R_i/N_{ii}$ for pattern i (where pattern i is the pattern containing only pieces of type i).

The same solution can be obtained by applying the simplex method to the model (without integrality constraints), where only the decision variables corresponding to the above single-item patterns are considered:

$$
\begin{array}{lccccccccccl}
\min & x_1 & + & x_2 & + & x_3 & + & x_4 & + & x_5 & & \\
s.t. & 5x_1 & & & & & & & & & \geq & 48 \\
 & & & 2x_2 & & & & & & & \geq & 35 \\
 & & & & & 2x_3 & & & & & \geq & 24 \\
 & & & & & & & 2x_4 & & & \geq & 10 \\
 & & & & & & & & & x_5 & \geq & 8 \\
 & x_1 & & x_2 & & x_3 & & x_4 & & x_5 & \geq & 0
\end{array}
$$

In fact, $x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 9.6 \\ 17.5 \\ 12.0 \\ 5.0 \\ 8.0 \end{bmatrix}$ corresponding to the basis $B = \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

Consider now a new possible pattern (number 6), containing one piece of type 1 and one piece of type 5. We ask ourselves: does the previous solution remain optimal if this new pattern is allowed? As we saw, we can answer a question like this by using duality. Recall that at every iteration the simplex method yields a feasible basic solution (corresponding to some basis B) for the primal problem and a dual solution (the multipliers $u^T = c_B^T B^{-1}$) that satisfy the complementary slackness conditions. (The dual solution will be feasible only at the last iteration.) The new pattern number 6 corresponds to including a new variable in the primal problem, with objective cost 1 (as each time pattern 6 is chosen, one rod is cut) and corresponding to the following column in the constraint matrix:

$$
A_6 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}
$$

This variables creates a new dual constraint. We then have to check if this new constraint is violated by the current dual solution u^T, i.e., if the reduced cost of the new variable with respect to basis B is negative. The new dual constraint is

$$
1u_1 + 0u_2 + 0u_3 + 0u_4 + 1u_5 \leq 1.
$$

The current dual solution associated to B is $u^T = c_B^T B^{-1} = \begin{bmatrix} 0.2 & 0.5 & 0.5 & 0.5 & 1 \end{bmatrix}$.

Since $0.2+1 = 1.2 > 1$, the new constraint is violated. This means that the current primal solution (in which the new variable is $x_6 = 0$) may not be optimal anymore (although it is still feasible). We can verify that the fact that the dual constraint is violated corresponds to the fact that the associated primal variable has negative reduced cost:

$$\bar{c}_6 = c_6 - u^T A_6 = 1 - \begin{bmatrix} 0.2 & 0.5 & 0.5 & 0.5 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = -0.2.$$

It is then convenient to let x_6 enter the basis. To do so, we modify the problem by inserting the new variable:

$$\begin{array}{lccccccccccccc}
\min & x_1 & + & x_2 & + & x_3 & + & x_4 & + & x_5 & + & x_6 & & \\
s.t. & 5x_1 & & & & & & & & & + & x_6 & \geq & 48 \\
 & & & 2x_2 & & & & & & & & & \geq & 35 \\
 & & & & & 2x_3 & & & & & & & \geq & 24 \\
 & & & & & & & 2x_4 & & & & & \geq & 10 \\
 & & & & & & & & & x_5 & + & x_6 & \geq & 8 \\
 & x_1 & & x_2 & & x_3 & & x_4 & & x_5 & & x_6 & \geq & 0
\end{array}$$

If this problem is solved with the simplex method, the optimal solution is found, but restricted only to patterns $1, \ldots, 6$. If a new pattern is available, one can decide whether this new pattern should be used or not by proceeding as above.

However, the problem is how to find a pattern (i.e., a variable; i.e, a column of the matrix) whose reduced cost is negative (i.e., it is convenient to include it in the formulation). Not only is the number of possible patterns exponentially large, but the patterns are not even known explicitly! The question then is:

> *Given a basic optimal solution for the problem in which only some variables are included, how can we find (if any exists) a variable with negative reduced cost (i.e., a constraint violated by the current dual solution)?*

This question can be transformed into an optimization problem: in order to see whether a variable with negative reduced cost exists, we can look for the minimum of the reduced costs of all possible variables and check whether this minimum is negative:

$$\begin{array}{ll}
\min & \bar{c} = 1 - u^T z \\
s.t. & z \text{ is a possible column of the constraint matrix.}
\end{array}$$

Recall that every column of the constraint matrix corresponds to a cutting pattern, and every entry of the column says how many pieces of a certain type are in that pattern. In order for z to be a possible column of the constraint matrix, the following condition must be satisfied:

$$z \in \mathbb{Z}_+^{|I|}$$
$$\sum_{i \in I} L_i z_i \leq W$$

Then the problem of finding a variable with negative reduced cost can be converted into the following integer linear programming problem:

$$
\begin{array}{lrcl}
\min & \bar{c} = 1 - \sum_{i \in I} u_i z_i & & \\
s.t. & \sum_{i \in I} L_i z_i & \leq & W \\
 & z & \in & \mathbb{Z}_+^{|I|}
\end{array}
$$

which is equivalent to the following (we just write the objective in maximization form and ignore the additive constant 1):

$$
\begin{array}{lrcl}
\max & \sum_{i \in I} u_i z_i & & \\
s.t. & \sum_{i \in I} L_i z_i & \leq & W \\
 & z & \in & \mathbb{Z}_+^{|I|}
\end{array}
$$

The coefficients z_i of a column with negative reduced cost can be found by solving the above integer knapsack problem.

In our example, if we start from the problem restricted to the five single-item patterns, the above problem reads as

$$
\begin{array}{lccccccccccc}
\max & 0.2z_1 & + & 0.5z_2 & + & 0.5z_3 & + & 0.5z_4 & + & z_5 & & \\
s.t. & 2.0z_1 & + & 4.5z_2 & + & 5.0z_3 & + & 5.5z_4 & + & 7.5z_5 & \leq & 11 \\
 & z_1 & & z_2 & & z_3 & & z_4 & & z_5 & \in & \mathbb{Z}_+
\end{array}
$$

and has the following optimal solution: $z^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \end{bmatrix}$. This correspond to the pattern called A_6 in the above discussion.

3 An algorithm for the one-dimensional cutting-stock problem

The procedure described above can be generalized to an algorithm for one-dimensional cutting-stock problems.

Problem 1 (One-dimensional cutting-stock problem)*: given*

- *a set of item types I,*
- *for every item type $i \in I$, its length L_i and the number of pieces to be produced R_i,*

- *the length W of the starting objects to be cut,*

find the minimum number of objects needed to satisfy the demand of all item types.

The problem can be modeled as follows:

$$\begin{array}{llll} \min & \sum_{j \in J} x_j & & \\ s.t. & \sum_{j \in J} N_{ij} x_j \geq & R_i & \forall \ i \in I \\ & x_j \in & \mathbb{Z}_+ & \forall \ j \in J \end{array}$$

where:

- J: set of all possible cutting patterns that can be used to obtain item types in I from the original objects of length W;
- N_{ij}: number of pieces of type $i \in I$ in the cutting pattern $j \in J$.
- x_j: number of original objects to be cut with pattern $j \in J$.

An algorithm for this problem is based on the solution of the continuous relaxation of the above model, i.e., the model obtained by replacing constraints $x_j \in \mathbb{Z}_+ \forall j \in J$ with constraints $x_j \in \mathbb{R}_+ \forall j \in J$.

Since $|J|$ can be so large as to make the enumeration of the patterns unpractical, the following algorithm can be used:

Step 0: initialization
Generate a subset of patterns J' such that the problem has a feasible solution (e.g., one can start with the $|I|$ single-item cutting patterns).

Step 1: solution of the master problem
Solve the master problem (restricted to the variables/patterns x_j with $j \in J'$)

$$\begin{array}{llll} \min & \sum_{j \in J'} x_j & & \\ s.t. & \sum_{j \in J'} N_{ij} x_j \geq & R_i & \forall \ i \in I \\ & x_j \in & \mathbb{R}_+ & \forall \ j \in J' \end{array}$$

thus obtaining a primal optimal solution x^* and a dual optimal solution u^* such that x^* and u^* satisfy the complementary slackness condition (this can be done with the simplex method).

Step 2: solution of the slave problem
Solve the following integer linear programming problem (called slave problem) with $|I|$ variables and one constraint:

$$\begin{array}{rrcl} \max & \sum_{i\in I} u_i^* z_i & & \\ s.t. & \sum_{i\in I} L_i z_i & \leq & W \\ & z_i & \in & \mathbb{Z}_+ \quad \forall \quad i \in I \end{array}$$

thus obtaining an optimal solution $z^* \in \mathbb{Z}_+^{|I|}$.

Step 3: optimality test
If $\sum_{i\in I} u_i^* z_i^* \leq 1$, then STOP: x^* is an *optimal solution* of the full continuous relaxation (including all patterns in J). Otherwise, *update the master problem* by including in J' the pattern γ defined by $N_{i\gamma} = z_i^*$ (this means that column z^* has to be included in the constraint matrix) and go to Step 1.

Finally, one has go from the optimal solution of the continuous relaxation to a heuristic (i.e., not necessarily optimal but hopefully good) solution of the original problem with integrality constraints. This can be done in at least two different ways:

- by rounding up the entries of x^* (this is a good choice if these entries are large: 765.3 is not very different from 766...); note that rounding down is not allowed, as we would create an infeasible integer solution;
- by applying an integer linear programming method (for instance the Branch-and-Bound) to the last master problem that was generated; this means solving the original problem (with integrality constraints) restricted to the "good" patterns (those in J') found in the above procedure.

4 Column generation methods for linear programming problems

The idea developed above for the one-dimensional cutting-stock problem can be applied to more general linear programming problems (without integer variables) whenever it is not possible or convenient to list explicitly all possible decision variables.

Consider the following problem:

$$\begin{array}{rrl} (P) & \min & c^T x \\ & \text{s.t.} & Ax = b \\ & & x \geq 0 \end{array}$$

where $A \in \mathbb{R}^{m \times n}$, and suppose that n (the number of variables) is very large, or that the n columns of A are not known explicitly. The dual problem is the following:

$$\begin{array}{lllll} (D) & \max & u^T b & & \\ & \text{s.t.} & u^T A & \leq & c^T \\ & & u & & free \end{array}$$

Step 0: initialization
Find explicitly a (small) subset of columns of A such that, if only these columns are considered, the problem has a feasible solution. Let $E \in \mathbb{R}^{m \times q} (q < n)$ denote this submatrix of A, and let x_E, c_E be the corresponding vectors of variables and costs in the objective function.

Step 1: solution of the Restricted Master Problem
Solve the following master problem:

$$\begin{array}{lll} (RMP) & \min & c_E^T x_E \\ & \text{s.t.} & E x_E = b \\ & & x_E \geq 0 \end{array}$$

thus obtaining a primal-dual pair of solutions x_E^M, u^M that are optimal for, respectively, (RMP) and its dual.

Theoretical remark. Consider the partition $A = \left[\begin{array}{c|c} E & H \end{array} \right]$ (Explicit, Hidden columns) and the corresponding $x = \left[\begin{array}{c} x_E \\ x_H \end{array} \right]$ e $c^T = \left[\begin{array}{c|c} c_E^T & c_H^T \end{array} \right]$. Note that the *extended* solution $x = \left[\begin{array}{c} x_E = x_E^M \\ x_H = 0^{n-q} \end{array} \right]$, is *feasible* for the initial problem (P) (as all constraints are satisfied). Furthermore, $u = u^M$ is a (not necessarily feasible) solution for (D): the number of entries of u is equal to the number of constraints of both (P) and (MP). Finally, u and x satisfy the complementary slackness conditions with respect to the initial pair (P)-(D). To see this, note that

$$(c^T - u^T A)x = ([c_E^T \mid c_H^T] - u^T [E \mid H]) \left[\begin{array}{c} x_E \\ x_H \end{array} \right] = (c_E^T - u^T E) x_E + (c_H^T - u^T H) \underbrace{x_H}_{=0} =$$

$$\underbrace{(c_E^T - (u^M)^T E) x_E^M}_{=0} + (c_H^T - (u^M)^T H) \cdot 0 = 0 \cdot x_E^M + 0 = 0,$$

as x_E^M and u^M satisfy the complementary slackness conditions (because they are optimal for (RMP) and its dual).

A pair of optimal solutions for (RMP) and its dual can be obtained *for instance* with the simplex method.

Step 2: solution of the slave problem (subproblem for the generation of a new column)
Find one or more vectors $z \in \mathbb{R}^m$ satisfying the following conditions:

(i) the entries of z are the coefficients in the constraint matrix of a variable x_j (i.e., z is a possible column A_j of A) whose cost is c_j;

(ii) $c_j - (u^M)^T z < 0$.

> *Theoretical remark.* The above conditions identify the existence of a constraint in the original dual problem (D) that is violated by the solution $u = u^M$. Note that (D) contains also all the constraints of the dual of (RMP), corresponding to the variables in x_E. These constraint are of course satisfied, as u^M is feasible for the dual of (RMP).

Step 3: optimality test
If no vector z as above exists, then **STOP**: $x = \begin{bmatrix} x_E^M \\ 0 \end{bmatrix}$ is an optimal solution of the initial problem (P).

> *Theoretical remark.* As we saw, $x = \begin{bmatrix} x_E = x_E^M \\ x_H = 0^{n-q} \end{bmatrix}$ and $u = u^M$ are a primal-dual pair of solutions for (P)-(D) satisfying the complementary slackness conditions. The fact that the slave problem is infeasible means that no constraint of (D) is violated, i.e., $u = u^M$ is feasible for (D). We then have pair of *feasible* solutions for (P)-(D) satisfying the complementary slackness conditions. By the strong duality theorem, x and u are optimal for (P) and (D).

Step 4: iteration
Update the master problem by including in matrix E one or more columns generated at Step 2; also update the corresponding x_E and c_E. Go to Step 1.

> *Theoretical remark.* As we saw, violated dual constraints correspond to variables with negative reduced cost; thus these variables are worth being included in the problem to improve the objective value.

5 Column generation methods: considerations on the implementation

5.1 The column generation subproblem

The critical part of the method is Step 2, i.e., generating the new columns. It is not reasonable to compute the reduced costs of all variables x_j for $j = 1, \ldots, n$, otherwise this procedure would reduce to the simplex method. In fact, n can be very large (as in the cutting-stock problem) or, for some reason, it mat be not possible or convenient to enumerate all decision variables.

> It is then necessary to device a specific column generation algorithm for each problem; only if such an algorithm exists (and is efficient), the method can be fully developed.

In the one-dimensional cutting stock problem we transformed the column generation subproblem into a resonable integer linear programming problem. In other cases, the computational effort required to solve the subproblem may be so high as to make the full procedure unpractical.

5.2 Convergence of the method

5.2.1 Feasibility and boundedness of the master problem

A column generation algorithm considers, at each iteration, a primal-dual pair of feasible solutions. In order for Step 1 to be able to find such a pair, the master problem needs to be *feasible and bounded*. At the first iteration feasibility can be achieved by taking any feasible solution for (P) and including in E only the columns corresponding to variables that take a strictly positive value in this solution. At the next iterations, if the method adds new variables, the new master problems will be feasible because the initial variables will still be included in the model. Moreover, to ensure boundedness, one can impose *box constraints*, i.e., constraints of the type $x_j \leq M$, $\forall\, j \in E$ (where M is a sufficiently large constant). In many cases such a value of M can be easily determined. (For instance, in the rod cutting problem it is easy to find a safe upper bound M on the number of rods needed.)

5.2.2 Convergence rate

The convergence of column generation methods is guaranteed by the theory of the simplex method, provided that the column generation subproblem can be solved exactly. However, from the practical point of view, convergence might be slow for several reasons (we only mention some of them below).

One issue is the following: if, at Step 4, a single variable is introduced, many iterations may be needed before including all variables needed in an optimal solution of the original problem. To overcome this problem, if possible one can find and include more than one new variable at every iteration.

Another issue is the fact that after some iterations problem (RMP) will contain a large number of variables, and therefore solving (RMP) may become very hard. One way of overcoming this problem is the creation of a pool of non-active variables among all the variables introduced so far. In other words, the variables whose value has been zero for several iterations can be eliminated from the model, but kept in a pool. However, when doing this, one has to ensure that the elimination of some variables does not make the problem infeasible. If this approach is adopted, at every iteration one can check if one of the columns already generated but currently removed has negative reduced cost; only if this is false, a new variable will be generated.

Some other problems, not covered here, are known as *instability*, *tailing-off*, *head-in* etc.: dealing with this aspects is fundamental for the implementation of efficient column generation methods (*stabilized column generation*).