1 Column Generation and the Cutting Stock Problem

In the linear programming approach to the traveling salesman problem we used the cutting plane approach. The cutting plane approach is appropriate when the number of variables is reasonably small and the number of constraints is big. Column generation applies to the dual situation where the number of constraints is small and the number of variables is big.

As a byproduct of the discussion we will see that a problem may have different ILP-formulations. The corresponding linear programming relaxation may behave very differently.

1.1 A General View at Column Generation (also called Pricing)

Column generation is a method of dealing with linear programs with an exponential number of variables. It is the dual of cut generation which deals with linear programs with an exponential number of constraints.

Consider a linear program and assume that x is a non-basic variable. Let a_x be the column of the constraint matrix corresponding to x and let c_x be the coefficient of x in the cost function.

We use A_B for the part of the constraint matrix involving the basis and x_B for the basis variables. Then

$$x_B = A_B^{-1}b - A_B^{-1}a_x x$$

and

$$z = c_B^T x_B + c_x x = c_B^T A_B^{-1} b + (c_x - c_B^T A_B^{-1} a_x)x.$$

It pays off to add x to the basis if the reduced cost $c_x - c_B^T A_B^{-1} a$ of the variable x is negative.

Pricing treats the column vector a_x as an unknown and asks itself whether there is a variable x with corresponding column a_x such that $c_x - c_B^T A_B^{-1} a_x < 0$.

1.2 The Cutting Stock Problem

The following problem arises in a sawing mill. We have an unlimited number of raws of length L and need to cut d_i rods of length l_i for $1 \leq i \leq k$. The goal is to minimize the number of raws used. Let $d = (d_1, \ldots, d_k)$ be the demand vector.

The following ILP-formulation is natural. Let $D = \sum_i d_i$. Then we certainly do not need more than D raws. Introduce integral variables x_{ij}, $1 \leq i \leq k$, $1 \leq j \leq D$, and integral variables z_j, $1 \leq j \leq D$. All variables are non-negative and the z_j are binary (only values zero and one): x_{ij} is the number of rods of length l_i cut from the j-th raw and z_j is one if the j-th raw is used. Our goal is to minimize $\sum_j z_j$ subject to the constraints $\sum_i l_i x_{ij} \leq L z_j$ for all j and $\sum_j x_{ij} = d_i$ for all i. Observe that the second class of constraints ensures that all demands are satisfied and that the first class of constraints guarantees that no raw is over-used. The problem with this ILP-formulation is that its linear programming relaxation gives very weak bounds.

Let $K = \sum_i d_i l_i / L$ be the total length of the rods demanded divided by the length of a raw. Then $\lceil K \rceil$ is certainly a lower bound on the number of raws required. The LP relaxation of the ILP-formulation above gives no better bound. We exhibit a solution to the LP with objective

value K. The idea is simply to use K rods (the first $\lfloor K \rfloor$ fully and the last one fractionally) and to distribute the demand evenly over the rods. Let $f = K - \lfloor K \rfloor$. We set

$$z_j = \begin{cases} 1 & \text{for } 1 \le j \le \lfloor K \rfloor \\ f & \text{for } j = \lfloor K \rfloor + 1 \\ 0 & \text{for } j > \lfloor K \rfloor + 1 \end{cases} \qquad \text{and for all } i,\ x_{ij} = \frac{d_i}{K} z_j$$

Then

$$\sum_i l_i x_{ij} = \sum_i l_i d_i z_j / K = L z_j \quad \text{for all } j$$

and

$$\sum_j x_{ij} = d_i / K(\lfloor K \rfloor + k) = (d_i/K)K = d_i \quad \text{for all } i$$

1.3 The Greedy Heuristik

There is a simple heuristic for the cutting stock problem. We consider the rods to be cut in arbitrary order. Assume that we have already produced some of the rods using some number of raws. We will have leftovers from the raws already in use and we will have fresh raws. When we have to cut a rod, we use one of the leftovers if there is a big enough leftover, and we will use a new raw otherwise.

Lemma 1 *The greedy strategy uses at most $\lfloor 2K+1 \rfloor$ raws.*

Proof: Number the raws in the order in which they are put into use and let G be the total number of raws used. For a raw j, let w_j be its unused part (= waste) and let u_j be its used part. Then

$$u_j + w_j = L \text{ for } 1 \le j \le G \quad \text{and} \quad \sum_j u_j = \sum_i d_i l_i = KL\ .$$

When the j-th rod ($j > 1$) is put into use, the leftovers from the preceding raws are to short to cut the rod under consideration. Thus

$$w_{j-1} \le u_j \quad \text{for } 1 < j \le G$$

Observe that this inequality holds when the j-th raw is put into use, and that wastes decrease and used parts increase over time. Thus the inequality holds at the end. Thus

$$GL = \sum_{1 \le j \le G} u_j + w_j \le w_G + 2 \sum_{1 \le j \le G} u_j \le L + 2KL$$

and hence $G \le 2K+1$. Since G is an integer we even have $G \le \lfloor 2K+1 \rfloor$. ∎

1.4 An Alternative ILP-Formulation

The alternative formulation uses a large number of variables. A cutting pattern is any vector $p = (a_1, \ldots, a_k)$ of non-negative integers such that $\sum_i a_i l_i \leq L$. There is in general an exponential number of cutting patterns. Trivial cutting patterns cut only rods of one kind, i.e., there is an i such that $a_i = \lfloor L/l_i \rfloor$ and $a_j = 0$ for $j \neq i$.

Let P be the set of all cutting patterns. For a pattern p let x_p the number of times the pattern p is used. The goal is to

$$\text{minimize } \sum_p x_p \text{ subject to } d = \sum_p p x_p,\ x_p \geq 0,\ x_p \text{ integral.}$$

As usual we obtain the LP relaxation by dropping the integrality constraint. The resulting LP has an exponential number of variables (but only k constraints). The advantage of the alternative LP over the natural LP is that it provides much better lower bounds for the underlying ILP.

We give a trivial example. Assume that we have raws of length 3 and want to cut five rods of length two. The lower bound tells us that we need at least four rods.

There is only one cutting pattern: Use a raw to produce a single rod of length one and hence we obtain the following ILP in a single variable.

$$\text{minimize } x_p \text{ subject to } 5 = x_p,\ x_p \geq 0,\ x_p \text{ integral.}$$

The ILP and LP relaxation have optimal value five. We invite the reader to work out a more complex example.

We need to discuss two issues:

- How to solve the linear program?
- How to go from an optimal solution to the LP to an optimal solution of the ILP.

1.5 Solving the LP

Let $P' \subset P$ be a subset of the cutting patterns. Initially, we might take P' as the set of all trivial cutting patterns. We solve the LP restricted to the variables in P'. The solution has a certain basis B. Let A_B be the matrix of cutting patterns in the basis. The cost vector is the all-one vector. Pricing leads to the following question:

Is there a cutting pattern, i.e., a vector $p = (a_1, \ldots, a_k)$ of non-negative integers with $\sum_i a_i l_i \leq L$, such that $1 - u^T p < 0$, where $u^T = \mathbf{1}^T A_B^{-1}$ and $\mathbf{1}$ is the all-one vector? This is a knapsack problem. We have a knapsack of capacity L and k different objects. The i-th object has weight a_i and value u_i. The question is whether we can fill the knapsack (using a_i objects of kind i) such that the capacity of the knapsack is not exceeded, i.e., $\sum_i a_i l_i \leq L$ and such that the value of the knapsack exceeds one, i.e., $u^T p = \sum_i u_i a_i > 1$.

Let us go through an example. The example is trivial but illustrative. We have raws of length 4. We want one rod of length 1 and one rod of length 3. We start with the trivial cutting patterns $(1,0)$ and $(0,1)$. Then $u^T = (1,1)$. So we ask whether there are a_1 and a_2 such that

$$1 \cdot a_1 + 3 \cdot a_2 \leq 4 \quad \text{and} \quad a_1 + a_2 > 1.$$

This has solutions $(2,0)$, $(3,0)$, $(4,0)$ and $(1,1)$.

The knapsack problem is NP-complete, but not strongly NP-complete. Dynamic programming usually works well for the knapsack problem.

1.6 Dynamic Programming for the Knapsack Problem

We assume that the l_i are integer. We fill a table T with index set $[0..L]$ such that $T[l]$ is the maximal value of a knapsack whose weight is at most l. We do first using only objects of type one, then objects of type one and two, ...

Suppose we have only one kind of object. There is not much choice we have: For $l < l_1$ the value is zero, i.e, $T[l] = 0$ for $l < l_1$ and for $l_1 \leq l \leq L$ we can add one item of value u_1 to a knapsack of weight at most $l - l_1$. Thus $T[l] = u_1 + T[l - l_1]$ for $l \geq l_1$.

Assume now that we considered the first $k-1$ items and now consider the k-th item. We can use it or not. Thus we should update the T-values according to the rule: For $l < l_k$ leave the T-value unchanged and for $l_k \leq l \leq L$ set $T[l] = \max(T[l], u_k + T[l - l_k])$.

It is important to consider the l's in increasing order (`for (int l = l[k], l <= L; l++)`) in the update step to account for the fact that we may take the k-th item any number of times. The initialization step is actually a special case of the update step. We could initialize T with the zero-vector and then start with $k = 1$. The running time of dynamic programming for the knapsack problem is $O(kL)$. This is pseudo-polynomial.

1.7 Solving the ILP

The solution of the LP gives us a lower bound for the optimal solution to the ILP. It also allows us to compute integer solutions.

The simplest way to obtain integer solutions is to round up the values of all variables. In this way all demands are certainly satisfied.

Alternatively, we could round down. Consider any basic variable x_p. We round it down and use the cutting pattern p, $\lfloor x_p \rfloor$ times. This will satisfy part of the demand (usually a large fraction of it). We satisfy the remaining demand using the greedy heuristic.

If we are lucky the LP lower bound tells us that the solution obtained in this way is optimal.

If not, we use branching. We take one of the fractional variables, say x, in the LP (one whose value, say β is close to half-integral) and generate two subproblems. One with the additional constraint $x \leq \lfloor \beta \rfloor$ and one with $x \geq \lceil \beta \rceil$.

2 General Cutting Plane Methods: Gomory Cuts

We show that ILPs can be solved by solving a series of linear programs.

Consider an integer linear program

$$\max \left\{ c^T x; Ax \leq b, x \geq 0, x \in \mathbb{N}^n \right\} \qquad \text{(ILP)}$$

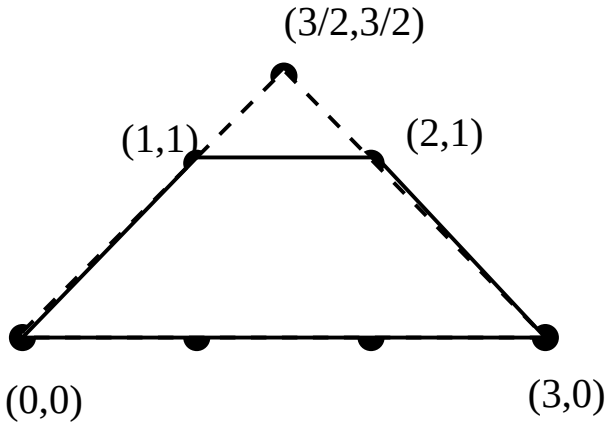

Figure 1: Consider the inequalities $y \leq x$, $y \leq 3-x$, $x \geq 0$, $y \geq 0$. The feasible region P is a triangle with vertices $(0,0)$, $(3,0)$, and $(3/2,3/2)$. The boundary of P is shown dashed. P contains the integral points $\{(0,0),(1,0),(2,0),(3,0),(1,1),(2,1)\}$. They comprise the feasible set F_I of the integer program. The boundary of the convex hull P_I of F_I is shown in bold. It is defined by the inequalities $y \leq x$, $y \leq 3-x$, $x \geq 0$, $y \geq 0$, and $y \leq 1$.

The *linear programming relaxation of ILP* is obtained by dropping the integrality constraint. We obtain the LP

$$\max\{c^T x; Ax \leq b, x \geq 0, x \in \mathbb{R}^n\} \tag{LP}$$

Clearly, the maximal objective value of the LP is at least the objective value of the ILP; the linear programming relaxation provides an upper bound.

Let $F_I = \{x; Ax \leq b, x \geq 0, x \in \mathbb{N}^n\}$ be the feasible set of ILP; observe that F_I is a discrete set. The feasible region of $P = \{x; Ax \leq b, x \geq 0, x \in \mathbb{R}^n\}$ of LP is a superset of F_I; it is a connected subset of $\mathbb{R}^n$. In principle, the optimization problem over F_I can be formulated as a linear program. The convex hull P_I of F_I is a convex set with vertices in F_I. We could describe P_I by a system of inequalities and then use this system to optimize, see Figure 4. The problem with this approach is that

- the description could be big (= it might require a very large number of constraints to define P_I,
- the description could be hard to find,
- it would be overkill to find a complete description of P_I. We only need a linear description in the vicinity of the optimal solution.

The idea of cutting plane methods is to add linear inequalities to the linear program as needed, namely to cut off fractional solutions (hence the name). The idea is simple. We call an inequality $a^T x \leq \delta$ valid for ILP if $a^T x \leq \delta$ for all $x \in F_I$. We have the following simple, but powerful lemma.

Lemma 2 *If a is integral and*

$$a^T x \leq \delta$$

is valid for ILP then

$$a^T x \leq \lfloor \delta \rfloor$$

is valid for ILP.

Proof: If $a^T x \leq \delta$ and $x \in F_I$ then $a^T x \in \mathbb{Z}$ and hence $a^T x \leq \lfloor \delta \rfloor$. ∎

We next discuss Gomory's method for finding valid inequalities of the form $a^T x \leq \delta$ with a integral and δ fractional.

Let B be an optimal basis of the LP (we assume for simplicity that the LP is bounded) and x^* be the corresponding optimal solution to the LP. If x^* is integral, it is also an optimal solution of ILP and we are done.

So assume that x^* is not integral. Then one of the basic variables, say x_i, has a fractional value. Consider the corresponding row of the dictionary

$$x_i = \bar{b}_i + \sum_{j \in N} \bar{a}_{ij} x_j \ . \tag{1}$$

$\bar{b}_i$ is a rational number which is not integral. For a rational number β let $\{\beta\} = \beta - \lfloor \beta \rfloor$ be the fractional part of β. Then $\{\bar{b}_i\} \neq 0$. We split equation 1 into is integral part and its fractional part and obtain

$$x_i - \lfloor \bar{b}_i \rfloor - \sum_{j \in N} \bar{a}_{ij} x_j = \{\bar{b}_i\} + \sum_{j \in N} \{\bar{a}_{ij}\} x_j \ .$$

For every feasible solution of our ILP the left hand side is an integer (since the variables are constrained to be integral) and hence the right hand side is integral for every feasible solution of our ILP. Thus

$$E = \{\bar{b}_i\} + \sum_{j \in N} \{\bar{a}_{ij}\} x_j \in \mathbb{Z}$$

for every feasible solution of the ILP.

Let us take a closer look at this expression. We have $\{\bar{b}_i\} > 0$ since $\bar{b}_i$ is fractional, we have $\{\bar{a}_{ij}\} \geq 0$ by the definition of fractional value and we have $x_j \geq 0$ since our variables are constrained to be non-negative. Thus $E > 0$ for every feasible solution of our ILP. Since E is known to be integral, we even have $E \geq 1$. Thus

$$\{\bar{b}_i\} + \sum_{j \in N} \{\bar{a}_{ij}\} x_j \geq 1$$

for every feasible solution of our ILP. We may therefore add this inequality to the ILP and also to its linear programming relaxation.

The addition of the new constraint has no effect on the ILP; we just argued that $E \geq 1$ for every feasible solution to the ILP. It does however have an effect on the linear programming relaxation. Observe that in our current solution, we have $x_j^* = 0$ for all $j \in N$ and hence the

current LP-solution is cut off by the new constraint. For this reason, the new constraint is called a cutting plane.

Let us look at an example. We have a ILP in two variables, say x and y, we are trying to maximize y and we have the constraints. $y \leq x$ and $y \leq 3 - x$. We add slack variables s and t and obtain the linear program $y + s - x = 0$ and $y + t + x = 3$. An optimal solution to the linear programming relaxation is $(x, y, s, t) = (3/2, 3/2, 0, 0)$. We use the fact that x has a fractional value to generate a cut. Solving for x in terms of the non-basic variables s and t yields

$$2x - s + t = 3 \quad \text{or} \quad x = \frac{3}{2} + \frac{1}{2}s - \frac{1}{2}t \ .$$

Splitting into integral and fractional part gives us

$$x - 1 - t = \frac{1}{2} + \frac{1}{2}s + \frac{1}{2}t \ .$$

The right hand side is integral and positive for every feasible solution of the ILP and hence we may generate the cut

$$\frac{1}{2} + \frac{1}{2}s + \frac{1}{2}t \geq 1 \quad \text{or} \quad s + t \geq 1$$

We may also interpret this constraint in terms of x and y. Adding our two original constraints gives $2y + s + t = 3$. Together with $s + t \geq 1$ this implies $y \leq 1$.

The technique for introducing new constraints just described was invented by Gomory in the late 50s and early 60s. It can be shown that if Gomory cuts are added in a careful way, the LP will have an integral optimal solution after a finite number of iterations, see [**?**, Section 23.8]. The number may be large, however. It must be large in the worst case, since integer linear programming is NP-complete.

I called this section general cuts, because it works for *every* integer linear program. For specific integer linear programs, one can also use problem-specific cuts. We will see an example below in the section on the Traveling Salesman Problem. Problem-Specific Cuts are usually much more effective than general cuts.

How does one resolve an LP after adding a cutting plane. We know it already (see Section **??**), but it is worthwhile to review it. We introduced the constraint $E \geq 1$ which is not satisfied by our current basis solution. We add a non-negative slack variable s, the equation $s = E - 1$, and add s to the basis. In this way we obtain a dual feasible basis (since we have not changed the cost function) which we can take as the initial basis for phase II of the simplex method.

3 The Traveling Salesman Problem

Go through the chapter on the Traveling Salesman Problem in the book by Cook et al [**?**].