The cutting stock problem

Thomas Loke

February 4, 2016

Suppose we have M orders, each respectively requiring q_j for $j = 1, \dots, M$ pieces. Suppose we have constructed a list of all possible combinations of patterns (say N of them, each with associated cost c_i), with each pattern used x_i for $i = 1, \dots, N$ times. The linear IP (integer program) is then:

$$\min \sum_{i=1}^{N} c_i x_i \tag{1}$$

$$\text{s.t.} \sum_{i=1}^{N} a_{ji} x_i \geq q_j \quad \forall\, j = 1, \dots, M \tag{2}$$

$$x_i \in \mathbb{Z}_{\geq 0} \qquad \forall\, i = 1, \dots, N \tag{3}$$

where a_{ji} is the number of times order j appears in pattern i.

Suppose that the stock material is of length L, and that order j is for a piece of length L_j. Then in pattern i, the amount of length used is $\bar{c}_i = \sum_{j=1}^{M} a_{ij} L_j$. If we want to minimize the number of utilised stock items, then we set $c_i = 1$ - otherwise, we minimize the total length wasted by setting $c_i = L - \bar{c}_i$.

The constraint on patterns is:

$$\bar{c}_i = \sum_{j=1}^{M} a_{ij} L_j \leq L \tag{4}$$

In order to minimize wastage (and to reduce problem size), we want to construct only maximal patterns, that is, patterns that cannot be filled with any additional orders. Mathematically, this gives the constraint:

$$L - \bar{c}_i = L - \sum_{j=1}^{M} a_{ij} L_j < L_{min} \tag{5}$$

where $L_{min} = \min(L_1, \dots, L_M)$

Putting these two constraints together gives the bound:

$$L - L_{min} < \bar{c}_i \leq L \tag{6}$$

or, expressed equivalently,

$$0 \leq L - \bar{c}_i < L_{min} \tag{7}$$

Suppose we have a given pattern P. Then we denote the number of times order j appears in the pattern P by $a_j(P)$. The pattern generation algorithm is as follows:

1. To minimize memory usage, sort the orders such that $L_1 \geq \ldots \geq L_M$, so $L_{min} = L_M$.
2. Initialize two sets $\bar{S} = \{P_0\}$ and $S = \emptyset$, where $\bar{S}$ is the set of non-maximal patterns, S is the set of maximal patterns, and P_0 is the starting (empty) pattern $P_0 = \{0, \ldots, 0\}$ (of length M) with associated cost $c(P_0) = L$, or written otherwise, $a_j(P_0) = 0$ for $j = 1, \ldots, M$. Now set $j = 1$.
3. For every pattern $P \in \bar{S}$ (at this fixed point in the algorithm):
   (a) Compute the maximal number of integer additions of order j to P, i.e. $a_j^{max}(P) = \lfloor c(P)/L_j \rfloor$.
   (b) If $j \neq M$, then do the following, otherwise skip to (c). Define new patterns P' with same parameters as P, with $a_j(P') \in \mathbb{Z}_{\geq 0}$ in the range $1 \leq a_j(P') \leq a_j^{max}(P) - 1$, and compute the corresponding costs $c(P') = c(P) - L_j a_j(P')$. Add all possible P' to the set $\bar{S}$.
   (c) Define the final new pattern P' with same parameters as P with $a_j(P') = a_j^{max}(P)$, and associated cost $c(P') = c(P) - L_j a_j(P')$. If $c(P') < L_{min}$, then add P' to S, otherwise add it to $\bar{S}$ (if $j \neq M$).
4. If $j \neq M$, then set $j \leftarrow j + 1$ and run step 3 again, otherwise terminate the algorithm with the set S as the set of maximal patterns and $N = |S|$.