1 Column generation formulation

We work with cutting patterns. For example, a raw roll of length \( r = 50 \) can be cut into many different patterns when the desired lengths are 14, 18, and 21:

<table>
<thead>
<tr>
<th>pattern</th>
<th>waste</th>
</tr>
</thead>
<tbody>
<tr>
<td>(3,0,0)</td>
<td>8</td>
</tr>
<tr>
<td>(0,2,0)</td>
<td>14</td>
</tr>
<tr>
<td>(0,0,2)</td>
<td>8</td>
</tr>
<tr>
<td>(1,2,0)</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Let \( a^j \) denote the \( j \)th cutting pattern, \( j = 1, 2, \ldots, P \). We have a very large number of possible patterns, so generate them as needed. Let \( x_j \) denote the number of raw rolls cut into pattern \( a^j \). The formulation is:

\[
\min_x \sum_{j=1}^p x_j \\
\text{subject to } \sum_{j=1}^p a^j x_j \geq b \\
x_j \geq 0, \text{ integer, } j = 1, \ldots, p
\]  

(CSP)

We solve the LP relaxation and then round at the end to get a feasible integer solution.

2 Getting an integer solution in the example problem

The solution to the LP relaxation is to take

\[
\frac{5}{4} \text{ of } a^4 = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}, \quad 7.5 \text{ of } a^2 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}, \quad \frac{27}{8} \text{ of } a^3 = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}.
\]

This uses \( \frac{5}{4} + 7.5 + \frac{27}{8} = \frac{97}{8} = 12 \frac{1}{8} \) rolls of raw material.

2.1 Rounding up

We can get a feasible integer solution by rounding up:

\[
2 \text{ of } a^4 = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}, \quad 8 \text{ of } a^2 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}, \quad 4 \text{ of } a^3 = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}, \quad \text{giving a total production of } \begin{bmatrix} 12 \\ 16 \\ 10 \end{bmatrix}.
\]
So, we can use 14 rolls of raw material to produce slightly more than the required (10, 15, 8).

Rounding up always gives a feasible solution. It will use at most \( m \) more rolls of raw material than the optimal solution. More precisely, let \( f_j \) be the fractional part of the number \( x_j \) of patterns \( j \). Let \( z_{LP} \) be the value of the LP relaxation. Let \( z_{RU} \) be the value of the solution obtained by rounding up. The value of the rounded up integer solution is

\[
z_{RU} = z_{LP} + \sum_{\text{basic vars } x_j} (1 - f_j) = z_{LP} + m - \sum_{\text{basic vars } x_j} f_j
\]

and \( z_{LP} \) gives a lower bound on the optimal value of the integer cutting stock problem.

### 2.2 Rounding down

We can also construct a solution by **rounding down** and heuristically covering the shortfall. Rounding down gives

1 of \( a^4 = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix} \), 7 of \( a^2 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \), 3 of \( a^3 = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} \), giving a total production of \( \begin{bmatrix} 9 \\ 14 \\ 7 \end{bmatrix} \).

These 11 rolls of raw material result in a shortfall of one unit for each of the three final lengths. This shortfall can be covered by using two patterns:

\[
\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},
\]

with an overall production plan of

1 of \( a^4 = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix} \), 7 of \( a^2 = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix} \), 3 of \( a^3 = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} \), 1 of \( \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \), 1 of \( \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \).

Thus we can meet the required demand using 13 rolls of raw material. This is **optimal**, since any feasible integer solution must use at least as many rolls as the solution to the LP relaxation, and it must use an integer number of rolls. So any feasible solution must use at least \( \lceil 12 \frac{1}{8} \rceil = 13 \) rolls of raw material.

### 2.3 Branch-and-price

We can solve the problem to integral optimality using branch-and-bound if we choose to not generate any additional patterns.

We can also embed the LP column generation problem in a branch-and-price algorithm to solve the integer program to optimality. Care is needed in the design of a branching routine, since adding simple bounds to the \( x_j \) variables changes the nature of the column generation subproblem.