Tutorial 10: Solving Cutting Stock Problem Using Column Generation Technique

GIAN Short Course on Optimization: Applications, Algorithms, and Computation

Devanand, Meenarli, Prashant, and Sven

IIT Bombay & Argonne National Laboratory

September 12-24, 2016
Linear Program in standard form:

Minimize $c^T x,$

Subject to $Ax = b,$

$x \geq 0.$

- $\mathcal{P}$ is the corresponding feasible set, matrix $A_{m \times n}$ has full row rank.
- $A_j$ is the $j^{th}$ column of the matrix $A$.
- $x$ be a ‘basic’ feasible solution and $B(1), \ldots, B(m)$ be the indices of basic variables.
- $NB = \{NB(1), \ldots, NB(n - m)\}$ be the indices of nonbasic variables.
- $B = \begin{bmatrix} A_{B(1)} & \cdots & A_{B(m)} \end{bmatrix}$ is basis matrix and $N = \begin{bmatrix} A_{NB(1)} & \cdots & A_{NB(n - m)} \end{bmatrix}$ be the nonbasis matrix.
Optimality conditions

- Vector \( x_N = (x_{NB(1)}, \ldots, x_{NB(n-m)}) \) for nonbasic variables is 0 and vector \( x_B = (x_B(1), \ldots, x_B(m)) \) of basic variables is obtained as

\[
Ax = [B \ N]^T \begin{bmatrix}
  x_B \\
  x_N
\end{bmatrix} = b,
\]

\[
Bx_B + Nx_N = b,
\]

\[
x_B = B^{-1} b.
\]

Consider moving away from \( x \) to \( x + \theta d^j \), \( \theta > 0 \) by selecting a nonbasic variable \( x_j, j \in N \).

**Algebraically**, \( d^j_j = 1 \) and \( d^j_i = 0, \forall i \in NB, i \neq j \).

\( x_B \) becomes \( x_B + \theta d^j_B \), where \( d^j_B = (d^j_{B(1)}, \ldots, d^j_{B(m)}) \)
For feasibility, \( A(x + \theta d^j) = b \) and

\[
\begin{align*}
Ad^j &= Bd^j_B + Nd^j_N = 0, \\
Bd^j_B + A_j &= 0, \\
d^j_B &= -B^{-1}A_j.
\end{align*}
\]

Objective value at the new point is \( c^T(x + \theta d^j) \) and per unit change along basic direction \( d^j \) (reduced cost of nonbasic variable \( x_j \)) is

\[
\bar{c}_j = c^T d^j = c_j - c_B^T B^{-1} A_j
\]

**Theorem (Optimality conditions)**

Consider a basic feasible solution \( x \) associated with a matrix \( B \), and let \( \bar{c} \) be the corresponding vector of reduced costs. If \( \bar{c} \geq 0 \), then \( x \) is optimal.
Motivation:

- Dantzig and Wolfe (1960) adapted it to LP with a decomposable structure.
- Other applications: Vehicle routing, crew scheduling, integer-constrained problems etc.
Column Generation

Recall

- number of nonzero variables (basic variables) is equal to the number of constraints.
- Hence even though the number of possible variables (columns) may be large, we only need a small subset of these (in basis B) in the optimal solution.

Crucial insight

- If a problem has many variables (or columns) but fewer constraints, work with a partial $A$ matrix.
Example: Cutting Stock Problem

Width of standard stock is 20 feet.

Demand is met by cutting up standard stocks into items of required widths (refer figure).

Objective is to minimize the number of standard stocks to meet the customer demands.
Problem description:
- Stock width $W_S$, and a set of items $\mathcal{I}$.
- Width of items denoted by $w_i$, and their demand $d_i$.
- Cost of using a stock per unit width is 1
- Set of cutting patterns $\mathcal{P}$
- $a_{ip}$: number of pieces of item $i \in \mathcal{I}$ cut in pattern $p \in \mathcal{P}$
- Minimize total cost (number of stocks used)

Decision variables:
- $x_p \in \mathcal{P}$: number of times a cutting pattern $p$ is used
Mathematical formulation (master)

Objective: Minimize total cost

\[
\text{Min } \sum_{i \in \mathcal{P}} x_p,
\]

Constraints

1. Demand of each item must be fulfilled

\[
\sum_{p \in \mathcal{P}} a_{ip} x_p \geq d_i, \quad \forall i \in \mathcal{I},
\]

2. Non-negativity and integrality constraints

\[
x_p \in \mathbb{Z}_+, \quad \forall p \in \mathcal{P}.
\]

Check:

\[
\sum_{i \in \mathcal{I}} a_{ip} w_i \leq W_S, \quad \forall p \in \mathcal{P},
\]
The Knapsack (sub) Problem

Problem description:
- Pick a new ‘pattern’ from $\mathcal{P}$ with most negative ‘reduced cost’
- ‘Value’ of item $v_i$, $i \in \mathcal{I}$ (multiplier of demand constraint)

Decision variables:
- $u_i \in \mathcal{I}$: number of times an item $i$ is cut in the (new) pattern

Objective: Minimize reduced cost

$$\text{Min } 1 - \sum_{i \in \mathcal{I}} u_i v_i$$

Constraints
1. The generated pattern must be valid
   $$\sum_{i \in \mathcal{I}} u_i w_i \leq W_S,$$
2. Non-negativity and integrality constraints
   $$u_i \in \mathbb{Z}_+ \quad \forall i \in \mathcal{I}.$$
Column Generation

repeat

Start with a set of ‘initial’ patterns ($m$) and solve the master problem.

Find the multipliers corresponding to the demand constraints: get $v_i$.

Solve the subproblem (knapsack) and obtain a new cutting pattern.

until the subproblem has a negative objective value;
AMPL Modeling Tip 1: Master and subproblem

Master problem

var Cut {PATTERNS} integer >= 0;  # stocks cut using a pattern

minimize Number:  # minimize total stock rolls
    sum {p in PATTERNS} Cut[p];

subject to Fill {i in ITEMS}:
    sum {p in PATTERNS} a[i,p] * Cut[p] >= demand[i];

Subproblem

var Use {ITEMS} integer >= 0;

minimize Reduced_Cost:
    1 - sum {i in ITEMS} price[i] * Use[i];

subj to Width_Limit:
    sum {i in ITEMS} i * Use[i] <= W_S;
AMPL Modeling Tip 2: Run File

model csp.mod; data csp.dat;
problem Cutting_Opt: Cut, Number, Fill;
problem Pattern_Gen: Use, Reduced_Cost, Width_Limit;

repeat {
solve Cutting_Opt;
let {i in WIDTHS} price[i] := Fill[i].dual;

solve Pattern_Gen;
if Reduced_Cost < -0.00001 then {
    let nPAT := nPAT + 1;
    let {i in WIDTHS} nbr[i,nPAT] := Use[i];
} else break;
};

See csp.mod, csp.dat and colgen.ampl