Gurobi Seminar 6
Cutting Stock Problem

Demetrios Papazaharias

UB INFORMS Student Chapter
Department of Industrial & Systems Engineering
University at Buffalo, SUNY

December 6, 2019

Welcome!

- Welcome to the sixth installment of our Gurobi series!
- In this workshop we will cover the cutting stock problem
- At the end of this workshop you will know how to implement column generation in Gurobi
- Materials for this workshop can be found here:

  `github.com/Dpapazaharias1/UB-INFORMS-Gurobi-Seminar`

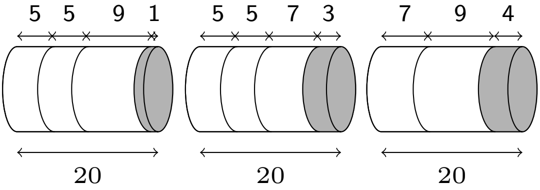
Problem Description

- The **cutting-stock problem** is the problem of cutting standard-sized pieces of stock material into pieces of specified sizes while minimizing material wasted.

- Consider a paper mill that has paper rolls with standard width 20 feet and customer demand:

Size	5-ft	7-ft	9-ft
Demand	25	20	15

- Some of the feasible cutting patterns to meet this demand are



Set Covering Formulation

- Parameters
  - W - fixed width of the standard stock material
  - q_j - the number of rolls demanded for item j
  - w_j - the width of item j
  - a_{ij} - number of times item j is cut in pattern i
- Decisions
  - x_i - number of times pattern i is used

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} x_i \\ \text{s.t.} \quad & \sum_{i=1}^{n} a_{ij} x_i \geq q_j \quad j = 1, \ldots, m \\ & x_i \in \mathbb{Z}_+ \quad i = 1, \ldots, n \end{aligned}$$

Why is Cutting Stock Hard?

- Each variable x_i is associated with one column in the constraint coefficient matrix $\mathbf{A}$.
- For cutting stock, how many columns (patterns) are there?

$$\frac{m!}{k!(m-k)!}$$

  Let k be the average number of items over all cutting patterns
- For larger problems considering this many variables explicitly is intractable
- We will use **column generation** (CG) to solve the LP relaxation of this problem

Column Generation

- The premise of CG is that most of the variables will be non-basic and have a value of zero in the optimal solution
- As a result, only a subset of variables need to be considered
- We begin with a small subset of variables and only add new variables if they have a potential to improve the objective function
- The procedure is split into a master problem and subproblem
  - The **master problem** is a restricted version of the original LP
  - The objective function of the **subproblem** is the reduced cost of the new variable with respect to the current dual of the master problem

Definitions

- Given two minimization problems

$$P_1 = \min\{f(x) : x \in X\}, \quad P_2 = \min\{g(x) : x \in Y\}$$

- We say that P_2 is a **relaxation** of P_1 if:
  1. $X \subset Y$
  2. $f(x) \geq g(x)$ ($\leq$ if $\max$)

  Example: LP relaxation of an IP!

- We say that P_2 is a **restriction** of P_1 if:
  1. $X \supset Y$
  2. $f(x) \leq g(x)$ ($\geq$ if $\max$)

Primal-Dual Relationship

- Consider the standard LP problem and its dual

$$(\mathcal{P}) \quad \begin{array}{ll} \min & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{array} \qquad\qquad (\mathcal{D}) \quad \begin{array}{ll} \max & \pi^\top \mathbf{b} \\ \text{s.t.} & \pi^\top \mathbf{A} \leq \mathbf{c} \end{array}$$

- The corresponding simplex tableau

x_B	x_N	RHS
B	N	b
$c_B^\top$	$c_N^\top$	0

$\Rightarrow$

x_B	x_N	RHS
I	$B^{-1}N$	b
0	$c_N^\top - c_B^\top B^{-1}N$	$-c_B^\top B^{-1}b$

- Recall from linear programming that $\pi^\top = c_B^\top B^{-1}$
- Reduced cost of a non basic variable: $r_j = c_j - c_B^\top B^{-1}a_j$. If $r_j < 0$ then we add x_j to the basis and improve the solution.
- This implies that there exists a constraint in the dual such that $\pi_j^\top a_j > c_j$ (No dual feasibility)
- Otherwise, $r_j \geq 0$ for all $j \in N$, and the solution is optimal.

Important Observations

- Since the master problem is a restriction on $(\mathcal{P})$, the dual is a relaxation on $(\mathcal{D})$
- Given the current dual solution, the subproblem finds the most violated constraint of $(\mathcal{D})$
- When we add a new variable to the master problem, we are adding a constraint to the dual!
- Given an optimal solution for the master problem, it must be optimal in the dual relaxation. If the dual solution is feasible in $\mathcal{D}$ then it must be optimal. Since you are optimal in $\mathcal{D}$, you must be optimal for $\mathcal{P}$.

CG for Cutting Stock

- Begin with a subset of cutting patterns $\mathcal{P}$, such that $|\mathcal{P}| = m$.
- Each pattern will be dedicated to roll width w_j
- For each width w_j the pattern produces $\left\lfloor \frac{W}{w_j} \right\rfloor$ rolls
- Consider the primal-dual for the LP relaxation the CSP

$$
\begin{aligned}
\min \quad & \sum_{i \in \mathcal{P}} x_i \\
\text{s.t.} \quad & \sum_{i \in \mathcal{P}} a_{ij} x_i \geq q_j \quad j = 1, \ldots, m \\
& x_i \geq 0 \quad i \in \mathcal{P}
\end{aligned}
\qquad
\begin{aligned}
\max \quad & \sum_{j=1}^{m} q_j \pi_j \\
\text{s.t.} \quad & \sum_{j=1}^{m} a_{ij} \pi_j \leq 1 \quad i \in \mathcal{P} \\
& \pi_j \geq 0 \quad j = 1, \ldots, m
\end{aligned}
$$

Generating Columns

- We wish to find a column in $\{1, \dots, n\} \setminus \mathcal{P}$ that can improve the optimal solution of (RMP)
- Given the optimal solution $\bar{\pi}$ the reduced cost column of pattern $i \in \{1, \dots, n\} \setminus \mathcal{P}$

$$1 - \sum_{j=1}^{m} a_{ij} \pi_j$$

- We want to add the column with most negative reduced cost. However we cannot list all of the cutting patterns. How can we generate the new column?

Subproblem

- Let y_j be a variable that represents a_{pj} for the new x_p
- Given the current dual solution $\bar{\pi}_j$. We wish to find the column (cutting pattern) $(y_1, \ldots, y_m)$ such that

$$\min \quad 1 - \sum_{j=1}^{m} \bar{\pi}_j y_j = 1 - \max \sum_{j=1}^{m} \bar{\pi}_j y_j$$

- However, we must ensure that y_j produces a feasible cutting pattern.

$$\sum_{j=1}^{m} w_j y_j \leq W, \quad y_j \in \mathbb{Z}_+$$

Each pattern must yield an integer number of rolls and the combined width of the rolls obtained is no larger than W

Subproblem

- Our subproblem is the following integer program

$$
\begin{aligned}
z^{SP} = \max \quad & \sum_{j=1}^{m} \bar{\pi}_j y_j \\
\text{s.t.} \quad & \sum_{j=1}^{m} w_j y_j \leq W \\
& y_j \in \mathbb{Z}_+ \quad j = 1, \ldots, m
\end{aligned}
$$

- Each y_j has some benefit $\bar{\pi}_j$, weight w_j and we cannot exceed limit W. What type of problem is this?

Subproblem

- Our subproblem is the following integer program

$$
\begin{aligned}
z^{SP} = \max \quad & \sum_{j=1}^{m} \bar{\pi}_j y_j \\
\text{s.t.} \quad & \sum_{j=1}^{m} w_j y_j \leq W \\
& y_j \in \mathbb{Z}_+ \quad j = 1, \ldots, m
\end{aligned}
$$

- Each y_j has some benefit $\bar{\pi}_j$, weight w_j and we cannot exceed limit W. What type of problem is this?

- This is a Knapsack Problem! An "easy" NP-Hard problem.

Subproblem

- Our subproblem is the following integer program

$$
\begin{aligned}
z^{SP} = \max \quad & \sum_{j=1}^{m} \bar{\pi}_j y_j \\
\text{s.t.} \quad & \sum_{j=1}^{m} w_j y_j \leq W \\
& y_j \in \mathbb{Z}_+ \quad j = 1, \ldots, m
\end{aligned}
$$

- Each y_j has some benefit $\bar{\pi}_j$, weight w_j and we cannot exceed limit W. What type of problem is this?

- This is a Knapsack Problem! An "easy" NP-Hard problem.

- Knapsack can be solved with an $O(mW)$ dynamic program

Updating the Master Problem

- If $z^{SP} \leq 1$, then $1 - \sum_{j=1}^{m} \bar{\pi}_j y_j^* \geq 0$. LP is optimal.
- If $z^{SP} > 1$, then $1 - \sum_{j=1}^{m} \bar{\pi}_j y_j^* < 0$. We have found the variable with the most negative reduced cost
- The new variable x_p is added to the model with coefficient $a_{pj} = y_j^*$ in the constraints:

$$\begin{aligned} \min \quad & \sum_{i \in \mathcal{P}} x_i + x_p \\ \text{s.t.} \quad & \sum_{i \in \mathcal{P}}^{n} a_{ij} x_i + a_{pj} x_p \geq q_j \quad j = 1, \ldots, m \\ & x_i,\ x_p \geq 0 \quad i \in \mathcal{P} \end{aligned}$$

Algorithm for CSP

- Start with initial columns: x_j cuts $\left\lfloor \frac{W}{w_j} \right\rfloor$ rolls of width w_j
- Solve the master problem and subproblem. While $z^{SP} < 0$:
  1. Solve the master problem to obtain optimal multipliers $\bar{\pi}$
  2. Identify a new column by solving the knapsack subproblem
  3. Add the new column x_p to the master problem
- CG only solves the LP relaxation. We need to solve the IP!
- Take the LP relaxation and change the variables to integer and solve the branch and bound. At each relaxation columns may be added to the relaxation
- This procedure is known as **branch-and-price**

Caveats

- Gurobi and CPLEX **cannot** do branch-and-price
- In practice many people do **early branching**. Namely, they solve the LP relaxation at the root node using CG and then solve the IP with only those generated columns.
  1. This is a heuristic procedure! You may never reach the optimal solution.
  2. Possible to have an infeasible IP solution
- In order to properly perform branch and price we must create our own branching scheme. Which is very difficult.
- In this workshop we will only focus on CG and early branching.