In the name of God

Part 3. ILOG CPLEX

3.5. Using Column Generation: A Cutting Stock Example

Spring 2010

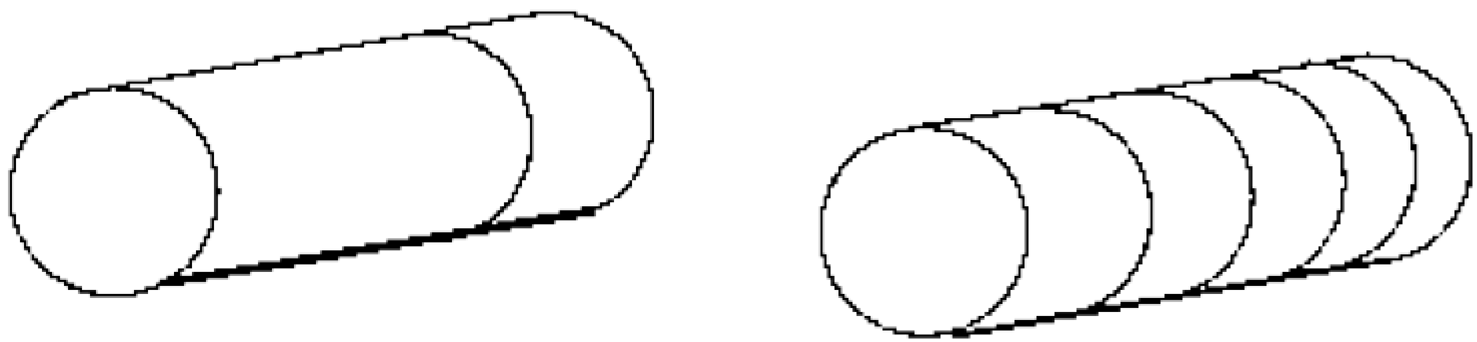
Instructor: Dr. Masoud Yaghini

Cutting Stock Problem

- **Rolls**
  - There is a supply of **rolls** of material of fixed length (the **stock**).
- **Strips**
  - **Strips** are cut from the rolls.
- **Patterns**
  - All the strips cut from one roll are known together as a **pattern**.

Cutting Stock Problem

- Two different patterns from a roll of stock:



- **Cutting stock problem**
  - The point is to use as few rolls of stock as possible to satisfy some specified demand of strips.

Cutting Stock Problem

- X_j : the number of times that pattern j will be used
- A_{ij} : the number of items i in pattern j
- d_i : the number of items i needed

$$\text{Minimize} \quad \sum_j X_j$$

s.t:

$$\sum_j \mathrm{A}_{ij} X_j \geq d_i, \quad \forall i$$

$$X_j \geq 0$$

Cutting Stock Problem

- To solve a cutting stock problem by column generation, start with a subproblem.
- Choose one pattern, lay it out on the stock, and cut as many items as possible, subject to the constraints of demand for that item and the width of the stock.
- This procedure will surely work in that it produces some answer (a feasible solution) to the problem, but it will not necessarily produce a satisfactory answer in this way since it probably uses too many rolls.

Cutting Stock Problem

- The subproblem is defined to identify the coefficients of a new column of the maser problem with **minimal reduced cost**.
- With π_i as the vector of the dual variables of the current solution of the master problem, the subproblem is defined like this:

$$\text{Minimize } 1 - \sum_i \pi_i A_i$$

$$s.t:$$

$$\sum_i A_i \leq W$$

Cutting Stock Problem

- W : the width of a roll of stock
- A_i : the variables of the subproblem
- Their solution values will be the coefficients of the new column to be added to the master model if a solution with a **negative objective function** is found for the subproblem.
- Consequently, the variables A_i must be nonnegative integers.

Representing the Data

- **static double RC_EPS**
  - If the objective value of the subproblem is nonnegative within the tolerance **RC_EPS**, then the application has proved that the current solution of the model is optimal within the given optimality tolerance (**RC_EPS**).
  - Otherwise, the application copies the solution of the current pattern generation problem into the array **newPatt** and uses that new pattern to build the next column to add to the model.
  - Then it repeats the procedure.

Representing the Data

- **static double rollWidth**
  - The width of a roll of stock.
  - This value is read from a file and represented by a numeric value, **rollWidth**.
- **static double[] size**
  - The widths of the ordered strips
  - Theses are read from a file and put into an array of numeric values, **size**.
- **static double[] amount**
  - The number of rolls ordered of each width
  - It is read from a file and put into an array of numeric values, **amount**.

Representing the Data

- **IloCplex cutSolver**
  - It is built to represent the master model.
- **IloCplex patSolver**
  - It represents the subproblem.
  - This is built to generate the new columns.

The Master Model

- Line 125
  - The master model is built
- Line 127
  - An objective **RollsUsed** is added to the master model
- Line 129-133
  - The application adds an array of constraints **Fill** to the model.
  - Constraint **Fill[i]** has a lower bound of **amount[i]** and an upper bound of **Double.MAX_VALUE**.

The Master Model

- Line 135-144
  - The master model is initialized with one variable for each size.
  - Each such variable represents the pattern of cutting a roll into as many strips of that size as possible.
  - These variables are stored as they are created in the array **Cut**
  - The variable **Cut[j]** will have an objective coefficient of 1 (one) and nonzero coefficient (**rollWidth/size[j]**) for constraint **Fill[j]**

The Master Model

- Line 146
  - Setting parameters

Column Generator

- Line 150
  - The subproblem model is built
- Line 152
  - An objective **ReducedCost** is added to the subproblem model
- Line 154
  - The pattern generator is defined by the integer variables in the array **Use**.
- Line 158
  - The only constraint added to subproblem model is a scalar product making sure that the patterns used do not exceed the width of rolls.

Column Generator

- Line 164
  - In the column generation loop, new variables will be added.
  - The variables will have coefficients defined by the solution vectors of the subproblem stored in the array **newPatt**.
- Line 172
  - The master problem is solved.
- Line 177
  - It copies the values of the dual solution into the array **price**.
- Line 178
  - It uses the array **price** to set objective coefficients in the model **patSolver**.

Column Generator

- Line 181
  - The right pattern generation problem is solved
- Line 184-187
  - If the objective value of the subproblem is greater than **-RC_EPS**, then the application has proved that the current solution of the model cutSolver is optimal within the given optimality tolerance (**RC_EPS**).

Column Generator

- Line 189
  - The application copies the solution of the current pattern generation problem into the array **newPatt**
- Line 191-197
  - They use that new pattern **newPatt** to build the next column to add to the model **cutSOlver**.
- Line 209-210
  - They de-allocate the models and algorithms after they are no longer in use

References

References

- ILOG CPLEX, **ILOG CPLEX User's Manual,** ILOG CPLEX, 2008.
- ILOG CPLEX, **ILOG CPLEX Java API Reference Manual,** ILOG CPLEX, 2008.

The End