Project Report

On

OPTIMISATION OF 1-D CUTTING STOCK PROBLEM

Submitted by

B Samhit Chowdary (170001015)

C Sai Dinesh (170001017)

Computer Science and Engineering

3rd year

Under the Guidance of

Dr. Kapil Ahuja

Department of Computer Science and Engineering

Indian Institute of Technology Indore

Autumn 2019

Introduction

- Cutting-stock problems can be classified in several ways. One way is the dimensionality of the cutting: one-dimensional (1D) problem; industrial applications of 1D occur when cutting pipes, cables, and steel bars. Two-dimensional (2D) problems are encountered in furniture, clothing and glass production. When either the master item or the required parts are irregular-shaped (a situation often encountered in the leather, textile, metals industries) this is referred to as the nesting problem.
- Not many three-dimensional (3D) applications involving cutting are known; however the closely related 3D packing problem has many industrial applications, such as packing objects into shipping containers.

Objectives

- To formulate the given problem into an Integer Linear Program.
- To study the approaches to the Integer Linear Program.
- To Implement the Delayed column generation approach to the above problem.

Problem Statement

- The Cutting problem is the problem of cutting standard sized pieces of stock material, such as paper rolls or sheet metal, into pieces of specified sizes while minimizing material waste. It is an optimization problem in mathematics that arise from application in industry.
- An Example:

  Suppose there is a factory which produces rolls of width 100 inches and the factory receives the following orders

Order Width	Quantity ordered
14	211
31	395
36	610
45	97

In this example each stock roll can be cut in various patterns for example : 7 rolls of width 14 each or 3 rolls of width 31 each or 2 rolls of width 31 and 1 roll of width 36...etc. Each of such combination is called a pattern and in this problem there are 37 distinct patterns possible. We observe that due to many patterns it is almost impossible to calculate this by hand therefore we solve it by formulating it into an integer linear program.

Formulation

Sets
I = set of order widths
J = set of patterns of cutting of rolls

Parameters
a_{ij} = number of rolls of width i cut in pattern j
b_i = demand for order width i

Decision Variables
X_j = number of rolls cut using pattern j

The objective of the cutting stock problem is to minimize the number of rolls cut subject to cutting enough rolls to satisfy the customer orders. Using the notation above, the problem can be formulated as follows:

Minimize $\sum_{j \in J} X_j$

subject to $\sum_{j \in J} a_{ij} X_j \geq b_i$, $\forall\ i \in I$

X_j integer.

Solution Approach

For this problem if no.of order widths is small then the no.of possible patterns is also small therefore it can be solved using standard **branch and bound algorithm**. But, if no.of order widths is very large then possible no.of patterns is also very high then it becomes very difficult to solve using branch and bound algorithm. In fact it will be very difficult to calculate the full set of patterns. Therefore we prefer to use different approach.

Delayed Column Generation Approach

- More generally if the stock rolls have width W and there are m different widths w_i (i = 1,....,m) in the order, we can generate all the patterns $a_j = [a_{1j} \ldots a_{mj}]^T$ of patterns consisting of a_{ij} rolls of width w_i that can be cut into the roll of width W, i.e., such that
$$\sum_{i=1}^{m} w_i a_{ij} \leq W .$$
- Note that $a_{ij} \in \mathbb{Z}^+ = \{0,\ 1,\ 2, \ldots..\}$ for all i,j.
- Conceptually assemble the columns a_j into a matrix A.
- Decision variables : $X_j \in \mathbb{N}_0$, the number of times pattern j is used (j = {1,...,n})
- Constraints : If there are b_i orders of width w_i, filling order implies $AX \geq b$, and due to the assumption that overproduction is waste, w.l.o.g ,
AX=b
- Objective : Minimize $\sum_{j=1}^{n} X_j$, the total number of stock rolls used.
- This yields Integer Programming model :

$$\text{(ICS) min } \sum_{j=1}^{n} X_j$$
$$\text{s.t } AX = b,$$
$$X \in \mathbb{Z}_+^n.$$

- The Linear Programming relaxation is given by

$$\text{(LCS) min } \sum_{j=1}^{n} X_j$$
$$\text{s.t } AX = b,$$
$$X \geq 0.$$

- The **dual** given by

$$\text{(DCS) max } b^T y$$
$$\text{s.t } A^T y \leq \overline{1}.$$

- The restricted pattern set :

$$\tilde{A} = \begin{bmatrix} \lfloor W/w_1 \rfloor & & 0 \\ & \ddots & \\ 0 & & \lfloor W/w_m \rfloor \end{bmatrix}$$

- Here we select only m patterns in each simplex iteration i.e., the columns corresponding to the basic variables. All the other variables (the non basic variables) are forced to zero.
- To solve the restricted sub problem , only need to solve the linear system

$$x = \tilde{A}^{-1} b .$$

- The optimum dual variables of the restricted LPM are given by $\bar{y} = \tilde{A}^{-T}\overline{1}$ (Using Complementarity Theorem)(lagrange multipliers of LCS in this case)
- If $\bar{y}$ is dual feasible (i.e., feasible for DCS), x is optimal for (LCS).
- Else there exists a pattern j (a column of full matrix A that has not yet been generated) corresponding to non basic variable X_j such that

$$\sum_{i=1}^{m} a_{ij}\bar{y}_i > 1 .$$

- Although pattern j has not yet been generated, we can find out whether or not it exists by solving knapsack problem

$$\text{(KS) max } \bar{y}^T p$$
$$\text{s.t } \sum_{i=1}^{m} w_i p_i \leq W$$
$$p \in \mathbb{Z}_+^m .$$

- If $\bar{y}^T p \leq 1$, then $\bar{y}$ is dual feasible i.e., matrix A is optimum, else put pattern p into matrix A and repeat the process.

- Now we use this matrix to solve the integer linear program to get the optimal set (which is quite easy to solve because matrix A now consists of only valid patterns)

  We implemented the above approach in matlab and gave input

  Question :

  Suppose there is a factory which produces rolls of width 100 inches and the factory receives the following orders

Order Width	Quantity ordered
14	211
31	395
36	610
45	97

Solution :

Initial matrix A:

	1	2	3	4
	7	0	0	0
	0	3	0	0
	0	0	2	0
	0	0	0	2

1 st iteration :

```
iteration 1
values of x
   30.1429
  131.6667
  305.0000
   48.5000

values of dual variables
    0.1429
    0.3333
    0.5000
    0.5000

Using 515.31 logs
new pattern
     2
     0
     2
     0
```

Matrix A after first iteration:

	1	2	3	4	5
1	7	0	0	0	2
2	0	3	0	0	0
3	0	0	2	0	2
4	0	0	0	2	0

2nd iteration :

```
iteration 2
values of x
         0
  131.6667
  199.5000
   48.5000
  105.5000

values of dual variables
         0
    0.3333
    0.5000
    0.5000

Using 485.167 logs
new pattern
     0
     2
     1
     0
```

Matrix A after 2nd iteration :

	1	2	3	4	5	6
1	7	0	0	0	2	0
2	0	3	0	0	0	2
3	0	0	2	0	2	1
4	0	0	0	2	0	0

3rd iteration :

```
iteration 3
values of x
         0
         0
  100.7500
   48.5000
  105.5000
  197.5000

values of dual variables
         0
    0.2500
    0.5000
    0.5000
```

After 3rd iteration max value of knapsack problem is less than 1.

Final answer :

```
Using 452.25 logs
Optimal solution uses 453 logs
Cut 100 logs with pattern
    2 cut(s) of length 36
    Waste of this pattern is 28
Cut 49 logs with pattern
    2 cut(s) of length 45
    Waste of this pattern is 10
Cut 106 logs with pattern
    2 cut(s) of length 14
    2 cut(s) of length 36
    Waste of this pattern is 0
Cut 198 logs with pattern
    2 cut(s) of length 31
    1 cut(s) of length 36
    Waste of this pattern is 2
Total waste in this problem is 130.
>>
```

MATLAB code has been uploaded on github

- https://github.com/Samhit-Chowdary/1-D-Cutting-Stock-Problem/blob/master/project.m

Conclusion :

- We have studied delayed column generation approach to solve this problem.
- We also learnt that if there are many variables of which many are active, then this approach is very helpful to obtain the optimum easily compared to the standard branch-bound algorithm.

References:

- https://neos-guide.org/content/cutting-stock-problem.
- https://www.youtube.com/watch?v=sUs70DshtWI.
- https://www.youtube.com/watch?v=NoiPrt4OsQA&t=2021s