Applications of Cutting Planes In Stochastic Integer Programs

A Project Report Submitted

in Partial Fulfilment of the Requirements

for the Degree of

BACHELOR OF TECHNOLOGY

in

Mathematics and Computing

by

Palash Goyal

(Roll No. 08012310)

to the

DEPARTMENT OF MATHEMATICS

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

GUWAHATI - 781039, INDIA

April 2012

CERTIFICATE

This is to certify that the work contained in this project report entitled "**Applications of Cutting Planes In Stochastic Integer Programs**" submitted by **Palash Goyal** (**Roll No.: 08012310**) to Indian Institute of Technology Guwahati towards partial requirement of **Bachelor of Technology** in Mathematics and Computing has been carried out by him under my supervision and that it has not been submitted elsewhere for the award of any degree.

Guwahati - 781 039 (Dr. Sriparna Bandopadhyay)
April 2012 Project Supervisor

ABSTRACT

The project explains about the generation of the cutting planes for Stochastic Programs for integer solutions. Initial chapter on Branch and Bound scheme is followed by an introduction to stochastic programming problem. The cutting plane technique for stochastic programming is in the form of feasibility and optimality cuts, and is known as the L-Shaped Method. Integer L-Shaped method is a further extension of the L-shaped method, which solves the stochastic integer programs, using the branch and bound scheme. The L-shaped method is used in the simple integer recourse of the stochastic programs, which results in integer solutions. The Production Problem has been solved using the simple integer recourse for the two-stage stochastic programs.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Integer Programming problems can be solved using enumeration, cutting planes and branching techniques. We will be first giving a short introduction about the Linear Programming (LP) problems before discussing the Integer Programming (IP) problems in this section.

1.1 Linear Programming Problem

A Linear Programming (LP) problem can be defined as:

$$\begin{aligned} \min z &= c^T x \\ \text{subject to: } Ax &= b \\ x &\geq 0 \end{aligned}$$

where $c, x \in \mathbb{R}^n, b \in \mathbb{R}^m$ *and* $A \in \mathbb{R}^{m \times n}$.

Whenever the LP relaxation problem gives a fractional solution, specific constraints on variables are added to the pre-existing ones to make the problem result in integral values for the variables. These constraints are known as cutting plane, or a set covering the cutting plane.

1.2 Integer Programming Problem

General Integer Programming (IP) problems can be modelled as

$$\begin{aligned} \min z &= c^T x \\ \text{subject to: } Ax &= b \\ x &\in Z^+. \end{aligned}$$

where $c \in \mathbb{R}^n, b \in \mathbb{R}^m, A \in \mathbb{R}^{m\times n}$; and $x \in \mathbb{Z}^+ = \{x : x \geq 0 \textit{ and } x \in \mathbb{Z}\}$.

1.2.1 Solution Techniques for Integer Programming problems

In the cutting plane technique of Gomory to solve Integer Progamming problems, the goal is to cut off the optimal solution to the linear programming relaxation and in doing so to cut off a part of the linear programming problems feasible region without cutting any of the feasible integer solutions.

Another method favored to solve integer programming problems is branching method. It splits up the problem into two different subproblems by choosing a variable that is not integer in the LP relaxation and allows this variable to first take the integer value greater than it and then the integer value lesser than it.

Example 1

The example focuses on the approach to find the cutting planes graphically.

The cutting plane is generated from the LP relaxation solution of the integer programming problem. Whenever the LP relaxation solution is not integral, then a cutting plane is generated by cutting off this solution. For simple two-dimensional examples, in most cases cutting planes can be generated graphically (refer [1]).

$$\begin{gathered}\min z = 6x_1 \text{ - } 5x_2 \\ \text{subject to: } 3x_1 + x_2 \leq 11 \\ -x_1 + 2x_2 \leq 5 \\ x_1,\ x_2 \in \mathbb{Z}^+.\end{gathered}$$

The optimal solution to the above LP relaxation is ($2\frac{3}{7}$,$3\frac{5}{7}$), as shown by Figure 1.1 or can be deduced from the simplex algorithm.

From the above solution we will find a plane such that:

i) It eliminates (cuts off) the current non-integer optimum solution of the LP relaxation.

ii) It is satisfied by all the integer feasible solutions to IP problems.

The solution of the example above is not an integral one, so more cutting planes need to be generated to serve the purpose. One of the cutting plane which can be added is:

$$x_1 + x_2 \leq 6.$$

This cutting plane can be obatined by taking $\frac{3}{7}$ of the first constraint and $\frac{2}{7}$ of the second and then rounding them appropriately.

This cutting plane cannot be defined as the strong cut, as it fails to remove a major portion of the feasible region. The cuts removing the largest portion

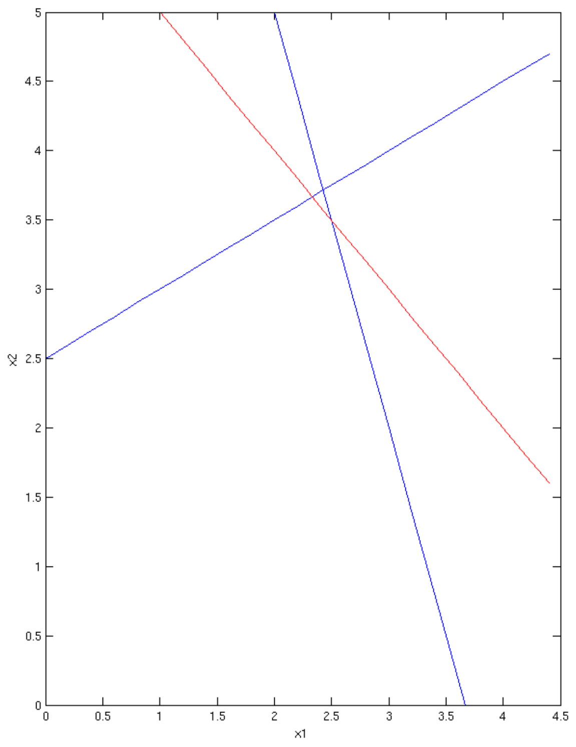

Figure 1.1: A cutting plane on a two dimensional integer programming problem

from the feasible region without removing any integer points are considered to be the best cuts. The best cutting plane is the cutting plane generated from the convex hull of the integer points.

Some of the other cutting planes which can be added to the above given example are given below :

$$x_1 + x_2 \leq 5$$
$$x_2 \leq 3$$
$$-x_1 + x_2 \leq 2$$
$$x_1 \leq 3.$$

The above cutting planes form the convex hull of the feasible integer points as shown in the Figure 1.2.

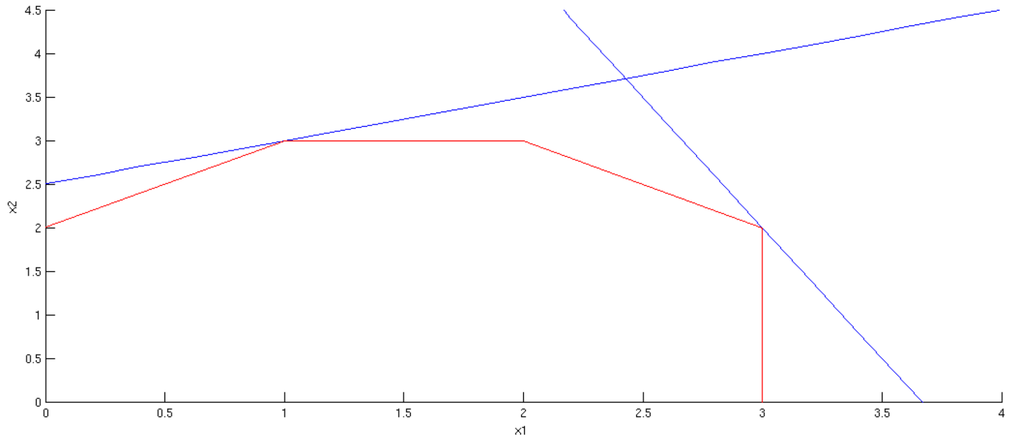

Figure 1.2: Convex Hull of the two dimensional integer programming problem

In the above example, the four constraints added later to the LP relaxation when solved using the simplex algorithm, the optimal solution to this new LP is also the solution to the IP. The above four constraints are satisfied because the simplex algorithm finds extreme point solutions to the linear programming problem and all of the extreme points are integers.

Chapter 2

Simplex

Given an Integer Programming problem, first the corresponding linear programming relaxation problem is solved. Using the optimal tables of the linear programming relaxation problem, Gomory suggested cutting planes for solving the interger programming problems. For the Gomory cutting planes, a prior discussion of the simplex algorithm to find the solutions to linear programming problems is given.

2.1 Primal Simplex

A discussion of the simplex algorithm is given in this subsection. Consider the following linear programming problem :

$$\begin{aligned} \min z \;&=\; c^T x \\ \text{subject to: } Ax &= b \qquad \text{(P)} \\ x &\geq 0. \end{aligned}$$

The dual of this is given by :

$$\begin{aligned} \max z &= b^T y \\ \text{subject to: } A^T y \leq c & \qquad \text{(D)} \\ y : \text{unrestricted in sign.} \end{aligned}$$

Theorem 2.1.1. Strong Duality Theorem: Suppose (P) is feasible with finite optimal value. Then (D) is feasible and there exists solution (x^*, y^*) such that x^* is primal feasible and y^* is dual feasible with $c^T x^* = b^T y^*$.

Given the primal formulation in (P) we can separate the variables into two groups, the basic and the nonbasic variables.

Let **B** be a basis matrix and without loosing generality let A = [B : N].

Therefore, we can write the original LP in the following form.

$$\begin{gathered} \min c_B^T x_B + c_N^T x_N \\ \text{subject to: } Bx_B + Nx_N = b \\ x_B, x_N \geq 0. \end{gathered}$$

Since B is a nonsingular matrix B^{-1} exists. By multiplying $Ax = b$ with B^{-1} we get :

$$\begin{gathered} \min c_B^T x_B + c_N^T x_N \\ \text{subject to: } x_B + B^{-1}Nx_N = B^{-1}b \\ x_B, x_N \geq 0. \end{gathered}$$

Definition 2.1.2. The reduced cost can be given as :

$$\bar{c}_N^T = c_N^T - c_B^T B^{-1} N.$$

When all of the reduced costs are non-negative then the table is optimal. If $\overline{c}_j < 0$ for some j, then a non-basic variable enters the basis and we choose the leaving variable according to the following minimum ratio test.

Definition 2.1.3. Minimum ratio test is :

$$min\{\frac{B^{-1}b_i}{\{B^{-1}N\}_{ij}} : \{B^{-1}N\}_{ij} > 0\}.$$

Let the new basis formed be B'. Repeat the above procedure till all the reduced costs $\overline{c}_j \geq 0$.

2.1.1 Algorithm

Simplex Algorithm

Step 1 Inputs x_B , c_N , B, N.

Step 2 Choose a negative reduced cost. This is the pivot column.

Step 3 Use the minimum ratio test to find the pivot row.

Step 4 Pivot in this row and column.

Step 5 If optimal, stop.

Step 6 Else go to *Step 3*.

2.2 Dual Simplex

The Algorithm for the Dual Simplex is as follows :

2.2.1 Algorithm

Basic Dual Simplex Algorithm

Step 1 Inputs x_B , c_N , B, N.

Step 2 Choose a negative $B^{-1}b$. This is the pivot row.

Step 3 Use the dual minimum ratio test to find the pivot column.

Step 4 Pivot in this row and column.

Step 5 If optimal, stop.

Step 6 Else go to *Step 3*.

Chapter 3

Gomory's Cutting Plane Algorithm

Gomory's cutting plane algorithm attempts to solve integer programming problems by cutting off the linear programming relaxation solution until an integer solution is found. As mentioned in the earlier sections, these cuts are generated from the rows of the optimal LP relaxation simplex table. If all the basic variables in the optimal solution are not integers then there exists a row i that has a fractional $B^{-1}b$ value (where B is the basis matrix, and $Bx_B = b$). From now on, we will be referring to $B^{-1}b$ as b. This row has the form:

$$x_{B_i} + \sum_{j \in NB} a_{ij} x_j = b_i \tag{3.1}$$

where NB is the set of non-basic variables in the simplex table. Rewriting (3.1) in terms of the fractional and integer parts yields:

$$x_{B_i} + \sum_{j \in NB} \lfloor a_{ij} \rfloor x_j + \sum_{j \in NB} f(a_{ij}) x_j = \lfloor b_i \rfloor + f(b_i) \tag{3.2}$$

where $f(a) = a - \lfloor a \rfloor$. Rearranging the terms, (3.2) becomes :

$$x_{B_i} + \sum_{j \in NB} \lfloor a_{ij} \rfloor x_j = \lfloor b_i \rfloor + f(b_i) - \sum_{j \in NB} f(a_{ij}) x_j \tag{3.3}$$

3.0.2 Valid Inequalities

Definition 3.0.1. *An inequality $\pi x + \sigma y \leq \pi_0$ is said to be valid for a feasible set of the integer program if $\pi \overline{x} + \sigma \overline{y} \leq \pi_0$ is satisfied for all $(\overline{x}, \overline{y})$ belonging to feasible set of the integer program* (refer [9]).

So, we can write (3.3) as an inequality :

$$x_{B_i} + \sum_{j \in NB} \lfloor a_{ij} \rfloor x_j \leq \lfloor b_i \rfloor + f(b_i). \tag{3.4}$$

If all the x_j's and x_{B_i} have integeral value then :

$$x_{B_i} + \sum_{j \in NB} \lfloor a_{ij} \rfloor x_j \leq \lfloor b_i \rfloor. \tag{3.5}$$

The optimal solution to the LP relaxation will violate constraint (3.5), unless b_i is integral.

On solving equation (3.1) for x_{B_i} we have :

$$x_{B_i} = b_i - \sum_{j \in NB} a_{ij} x_j. \tag{3.6}$$

If this expression of x_{B_i} is substituted in (3.5) then the following inequality is valid :

$$\sum_{j \in NB} f(a_{ij})x_j \geq f(b_i). \tag{3.7}$$

We will now have a look at the method for generating strong valid inequalities, which dominate the above mentioned valid inequalities.

3.0.3 Strong valid inequalities

The original Gomory fractional cuts can be strengthened in a couple of different ways. The first way that will be discussed is from [5] and [4].
"For any integer t, the cut

$$\sum_{j \in NB} f(ta_{ij})x_j \geq f(tb_i) \tag{3.8}$$

is satisfied by all non-negative integer solutions to (3.6) and therefore by all solutions to (IP). Also, if $f(b_i) < \frac{1}{2}$, t is positive and $\frac{1}{2} \leq tf(b_i) < 1$, then (3.8) dominates (3.7) (refer [5])."

Dominant inequalities are valid inequalities that cut off more of the feasible region than the other inequalities (refer [7]).

Definition 3.0.2. *If $\pi x \leq \pi_0$ and $\mu x \leq \mu_0$ are two valid inequalities, then $\pi x \leq \pi_0$ dominates $\mu x \leq \mu_0$ if there exists $u > 0$ such that $\pi > u\mu$ and $\pi_0 \leq u\mu_0$, and $(\pi, \pi_0) \neq (u\mu, u\mu_0)$* (refer [9]).

The second technique listed in [5] and [6] is a strengthening cut that can be generated by looking at the mixed integer Gomory cuts. One of the theorem found in [7] states,

Theorem 3.0.3. *If* $X = \{(x,y) \in \mathbb{R}^1_+ \times \mathbb{Z}^1 : y \leq b + x\}$*, and* $f = b - \lfloor b \rfloor > 0$*, the inequality*

$$y \leq \lfloor b \rfloor + \frac{x}{1-f} \tag{3.9}$$

is valid for X*.*

To refer the integer parts of the Gomory cuts in [7], like before there must be a row i which has a fractional right hand side b.

$$x_{B_i} + \sum_{j \in NB} a_{ij} x_j = b_i. \tag{3.10}$$

We can break up the nonbasic variables into two sets

$$x_{B_i} + \sum_{f_j \leq f_0} a_{ij} x_j + \sum_{f_j > f_0} a_{ij} x_j = b_i \tag{3.11}$$

where $f_j = f(a_{ij})$ and $f_0 = f(b_i)$. We can rewrite equation (3.11) as

$$x_{B_i} + \sum_{f_j \leq f_0} \lfloor a_{ij} \rfloor x_j + \sum_{f_j \leq f_0} f_j x_j + \sum_{f_j > f_0} \lceil a_{ij} \rceil x_j - \sum_{f_j > f_0} (1 - f_j) x_j = b_i. \tag{3.12}$$

Rearranging the terms by putting the integer parts on the left and the other parts on the right.

$$x_{B_i} + \sum_{f_j \leq f_0} \lfloor a_{ij} \rfloor x_j + \sum_{f_j > f_0} \lceil a_{ij} \rceil x_j = b_i - \sum_{f_j \leq f_0} f_j x_j + \sum_{f_j > f_0} (1 - f_j) x_j. \tag{3.13}$$

The above equation can be written as an inequality

$$x_{B_i} + \sum_{f_j \leq f_0} \lfloor a_{ij} \rfloor x_j + \sum_{f_j > f_0} \lceil a_{ij} \rceil x_j \leq b_i + \sum_{f_j > f_0} (1 - f_j) x_j \tag{3.14}$$

Using Theorem 3.0.3, we obtain the valid inequality,

$$x_{B_i} + \sum_{f_j \le f_0} \lfloor a_{ij} \rfloor x_j + \sum_{f_j > f_0} \lceil a_{ij} \rceil x_j \le \lfloor b_i \rfloor + \sum_{f_j > f_0} \frac{1-f_j}{1-f_0} x_j. \tag{3.15}$$

Substituting x_{B_i} from equation (3.11) into inequality (3.15) and simplifying results

$$f_0 \le \sum_{f_j \le f_0} f_j x_j + \sum_{f_j > f_0} \frac{f_0(1-f_j)}{1-f_0} x_j. \tag{3.16}$$

Notice that when $f_j \le f_0$ then:

$$f_j \le f_0 = f_0 \frac{1-f_0}{1-f_0} \le f_0 \frac{1-f_j}{1-f_0} \tag{3.17}$$

and when $f_j > f_0$ then:

$$f_0 \frac{1-f_j}{1-f_0} \le f_0 \frac{1-f_0}{1-f_0} = f_0 < f_j. \tag{3.18}$$

Thus, we can rewrite inequality (3.16) as

$$f_0 \le \sum_{j \in NB} min\{f_j, f_0 \frac{1-f_j}{1-f_0}\} x_j \tag{3.19}$$

is valid and (3.19) is stronger than (3.7) [5].

One might presume that we can use the first strengthening technique on this inequality. However, this is not so because all of the coefficients are lesser than f_0 . Refer (3.17) and (3.18).

On the other hand, we can use the second strengthening (3.19) on the first strengthening technique (3.8). This gives :

$$\sum_{j \in NB} min\{f(ta_{ij}), f(tb_i)\frac{1 - f(ta_{ij})}{1 - f(tb_i)}\}x_j \geq f(tb_i). \tag{3.20}$$

This theory also works on the objective function row. As $c^T x = -z$, so if $-z$ is fractional then the Gomory cutting plane for the objective function row can be added to the table. Therefore the same formulas will work for the objective function.

3.0.4 Algorithm

Gomory Cutting Plane Algorithm

Step 1 Find the Simplex Table.

Step 2 Find the Gomory Cutting Planes associated with each row that has a fractional right hand side.

Step 3 Add these cutting planes to the Simplex table maintaining primal feasibility.

Step 4 Use the Dual Simplex Algorithm to find a solution to the new LP.

Step 5 If optimal, stop.

Step 6 Else go to *Step 2*.

Given below is an example illustrating the Gomory's cutting plane algorithm.

Example 6

$$\min z = -2x_1 - 3x_2$$
$$\text{subject to : } x_1 - x_2 \leq 1$$
$$4x_1 + x_2 \leq 28$$
$$x_1 + 4x_2 \leq 27$$
$$x_1,\ x_2 \in \mathbb{Z}^+ \ .$$

Let x_3, x_4, x_5 be the slack variables corresponding to the above three constraints. So, the simplex table of the linear programming relaxation is :

	x_1	x_2	x_3	x_4	x_5
0	-2	-3	0	0	0
1	1	-1	1	0	0
28	4	1	0	1	0
27	1	4	0	0	1

The optimal form of this table is :

	x_1	x_2	x_3	x_4	x_5
$\frac{82}{3}$	0	0	0	$\frac{1}{3}$	$\frac{2}{3}$
$\frac{2}{3}$	0	0	1	$-\frac{1}{3}$	$\frac{1}{3}$
$\frac{17}{3}$	1	0	0	$\frac{4}{15}$	$-\frac{1}{15}$
$\frac{16}{3}$	0	1	0	$-\frac{1}{15}$	$\frac{4}{15}$

The constraint corresponding to the third row is :

$$x_2 - \frac{1}{15}x_4 + \frac{4}{15}x_5 = \frac{16}{3}. \tag{3.21}$$

Applying the rounding technique of equation (3.5) on the third row constraint, we obtain the floor of all of the coefficients and have this valid inequality :

$$x_2 - x_4 \leq 5 \tag{3.22}$$

We have rounded the equation (3.21) by using the fractional parts as described in (3.7) :

$$\frac{14}{15}x_4 + \frac{4}{15}x_5 \geq \frac{1}{3}. \tag{3.23}$$

On putting the inequalities (3.22) and (3.23) in terms of the original variables x_1 and x_2, the inequality (3.22) becomes:

$$\begin{gathered} x_2 - x_4 \leq 5 \\ x_2 - (28 - 4x_1 - x_2) \leq 5 \\ 4x_1 + 2x_2 \leq 33 \end{gathered}$$

and the inequality (3.23) becomes :

$$\begin{gathered} \tfrac{14}{15}x_4 + \tfrac{4}{15}x_5 \geq \tfrac{1}{3} \\ \tfrac{14}{15}(28 - 4x_1 - x_2) + \tfrac{4}{15}(27 - x_1 - 4x_2) \geq \tfrac{1}{3} \\ -4x_1 - 2x_2 + \tfrac{100}{3} \geq \tfrac{1}{3} \\ 4x_1 + 2x_2 \leq 33. \end{gathered}$$

In the inequality (3.23) the right hand side $b_3 = \frac{1}{3}$ fits the criteria given in the equation (3.8). So we can multiply (3.23) by t = 2 and obtain a stronger valid inequality.

$$2.(\frac{14}{15}x_4 + \frac{4}{15}x_5) \geq 2.(\frac{1}{3}) \tag{3.24}$$

$$\frac{28}{15}x_4 + \frac{8}{15}x_5 \geq \frac{2}{3} \tag{3.25}$$

On rounding the equation (3.25), we have get :

$$\frac{13}{15}x_4 + \frac{8}{15} \geq \frac{2}{3}. \tag{3.26}$$

The above inequality (3.26) comes out to be stronger than the (3.23) one, because (3.25) is equivalent to (3.23) but (3.26) is harder to satisy than (3.25) as it has been rounded. The inequality (3.26) in terms of the original variables comes out to be the following:

$$\frac{13}{15}x_4 + \frac{8}{15}x_5 \geq \frac{2}{3} \tag{3.27}$$

$$\frac{13}{15}(28 - 4x_1 - x_2) + \frac{8}{15}(27 - x_1 - 4x_2) \geq \frac{2}{3} \tag{3.28}$$

$$4x_1 + 3x_2 \leq 38. \tag{3.29}$$

When we make use of one more strengthening technique mentioned in (3.19), the inequality (3.23) takes the form :

$$min\{\frac{14}{15}, \frac{1}{3}.\frac{\frac{1}{15}}{\frac{2}{3}}\}x_4 + min\{\frac{4}{15}, \frac{1}{3}.\frac{\frac{11}{15}}{\frac{2}{3}}\}x_5 \geq \frac{1}{3} \tag{3.30}$$

$$min\{\frac{14}{15}, \frac{1}{30}\}x_4 + min\{\frac{4}{15}, \frac{11}{30}\}x_5 \geq \frac{1}{3} \tag{3.31}$$

$$\frac{1}{30}x_4 + \frac{4}{15}x_5 \geq \frac{1}{3}. \tag{3.32}$$

The above inequality (3.32) is stronger as compared to (3.23) as the coefficeint of x_4 is less, and this makes the inequality (3.32) more harder to satisfy.

On putting the inequality (3.32) in terms of the original variables, we have get:

$$\frac{1}{30}x_4 + \frac{4}{15}x_5 \geq \frac{1}{3} \tag{3.33}$$

$$\frac{1}{30}(28 - 4x_1 - x_2) + \frac{4}{15}(27 - x_1 - 4x_2) \geq \frac{1}{3} \tag{3.34}$$

$$4x_1 + 11x_2 \leq 78. \tag{3.35}$$

Similarly, we have generated more cuts using the above procedure and the final integral values obtained are $x_1 = 5$ and $x_2 = 5$. So, the objective function has value:

$$\begin{aligned} z &= -2x_1 - 3x_2 \\ &= -10 - 15 \\ &= -25. \end{aligned}$$

Chapter 4

Example

The following section includes the results obtained from the MATLAB codes.

4.1 Example

This part shows the MATLAB result for the Gomory cutting-plane Algorithm.

The problem considered is as follows:

$$\max \qquad z = 3x_1 + 4x_2$$

subject to :

$$3x_1 - x_2 \leq 12$$
$$3x_1 + 11x_2 \leq 66$$
$$x_1, x_2 \geq 0$$

The code displays the initial and the final tables of the simplex algorithm. Moreover, all the tables generated during the execution of the dual simplex method are also included in the output.

The results for Gomory Cutting-Plane Algorithm has been obatined by running the *cpa* code which is making use of the two functions *fractp* and *simplex*. The *fractp* code is computing the fractional parts of real numbers that are stored in a matrix, and the *simplex* code is finding an optimal solution to the problem without the integral restrictions imposed on its proposed variables.

On running the *cpa* code, we get the following tables :

```
      ------------------------------------------------

                Table of the Simplex Algorithm

      ------------------------------------------------
                     Initial table


A =

     3    -1     1     0    12

     3    11     0     1    66

    -3    -4     0     0     0


 Press any key to continue ...


                          Final table

      ------------------------------------------------
```

```
A =

    1.0000         0    0.3056    0.0278    5.5000

         0    1.0000   -0.0833    0.0833    4.5000

         0         0    0.5833    0.4167   34.5000

 Press any key to continue ...


      ------------------------------------------------


              Tables of the Dual Simplex Method

      ------------------------------------------------
    pivot row-> 3 pivot column-> 3


                         Table 1


A =


    1.0000         0    0.3056    0.0278         0    5.5000

         0    1.0000   -0.0833    0.0833         0    4.5000

         0         0   -0.3056   -0.0278    1.0000   -0.5000

         0         0    0.5833    0.4167         0   34.5000


 Press any key to continue ...
```

```
    pivot row-> 4 pivot column-> 5
                          Table 2


A =

    1.0000         0         0         0    1.0000         0    5.0000

         0    1.0000         0    0.0909   -0.2727         0    4.6364

         0         0    1.0000    0.0909   -3.2727         0    1.6364

         0         0         0   -0.0909   -0.7273    1.0000   -0.6364

         0         0         0    0.3636    1.9091         0   33.5455

 Press any key to continue ...



    pivot row-> 5 pivot column-> 4
                          Table 3


A =

1.0000         0         0   -0.1250         0    1.3750         0    4.1250

     0    1.0000         0    0.1250         0   -0.3750         0    4.8750

     0         0    1.0000    0.5000         0   -4.5000         0    4.5000

     0         0         0    0.1250    1.0000   -1.3750         0    0.8750

     0         0         0   -0.1250         0   -0.6250    1.0000   -0.8750

     0         0         0    0.1250         0    2.6250         0   31.8750

 Press any key to continue ...
```

```
     ------------------------------------------------
                       Final table

     ------------------------------------------------


A =

1.0000         0         0         0         0    2.0000   -1.0000    5.0000

     0    1.0000         0         0         0   -1.0000    1.0000    4.0000

     0         0    1.0000         0         0   -7.0000    4.0000    1.0000

     0         0         0         0    1.0000   -2.0000    1.0000         0

     0         0         0    1.0000         0    5.0000   -8.0000    7.0000

     0         0         0         0         0    2.0000    1.0000   31.0000

 Press any key to continue ...

     ------------------------------------------------
            Problem has a finite optimal solution

     ------------------------------------------------
 Values of the proposed variables:


 x(1)= 5

 x(2)= 4


 Objective value at the optimal point:


 z= 31.000000
```

Chapter 5

Branch and Bound Method

The Branch and Bound method is widely used to solve both pure and mixed integer linear programming problems. Sometimes a few or all the variables of an integer linear programming problem are constrained by their upper or lower bounds.

The branch and bound method first divides (branches) the feasible region into smaller subsets and then examines each of them successively until a feasible solution that gives the optimal value of the objective function is obtained.

Let the given integer linear programming problem (IPP) be :

$$\begin{gathered} \max \quad z = c^T x \\ \text{subject to} \quad Ax \leq b, \\ x \geq 0, \quad integers. \end{gathered}$$

In this method, we first solve the problem by ignoring the integer constraints :

(i) If the solution is integer, the current solution is optimum for the given

IPP.

(ii) If the solution is not an integer, say one of the variable x_r is not an integer, where x_r^* is the smallest integer less than or equal to x_r, then any feasible integer value of x_r must satisfy one of the two conditions :

a) $x_r \leq x_r^*$, or

b) $x_r \geq x_r^* + 1$.

By adding these two conditions separately to the given linear programming problem we form different subproblems :

$$\begin{array}{ll} \text{Subproblem 1} & \text{Subproblem 2} \\ \max \quad z = c^T x & \max \quad z = c^T x \\ \text{s.t.} \quad Ax \leq b, & \text{s.t.} \quad Ax \leq b, \\ \quad x_r \leq x_r^*, & \quad x_r \geq x_r^* + 1, \\ \quad x \geq 0. & \quad x \geq 0. \end{array}$$

Here we have branched or partitioned the original problem into two subproblems. Each of these subproblems is then solved separately as a linear programming problem. If any subproblem yields an optimum integer solution it is not further branched. But if any subproblem yields a non-integer solution it is further branched into two subproblems. This branching process is continued until each problem terminates with either an integer optimal solution or there is evidence that it cannot yield a better solution. The integer valued solution among all the subproblems which gives the most optimal value of the objective function is then selected as the optimum solution.

The subproblem is said to be fathomed and is dropped from further consideration in one of the following cases:

1. The subproblem has an optimal solution with $z \leq z^*$ where z^* is the current best solution;

2. The subproblem has no feasible solution;

3. The subproblem has an optimal solution that has all integer values.

5.0.1 Example

$$\begin{aligned} \max \quad & z = 5x_1 + 8x_2 \\ \text{s.t.} \quad & x_1 + x_2 \leq 6, \\ & 5x_1 + 9x_2 \leq 45, \\ & x_1, x_2 \geq 0, \qquad \textit{integer}. \end{aligned}$$

Optimal solution of the LP relaxation of the above problem is $(x_1, x_2) = (2.25, 3.75)$ with the optimal value of objective function as $z = 41.25$.

The optimal value of the LP relaxation is an upper bound for the optimal value of the integer program. In this case, 41.25 is an upper bound for the optimal value, z_{opt}, of the integer programming problem.

We choose the variable x_2 that is fractional in the optimal solution for the LP relaxation.

Every feasible solution of the integer programming problem must have either $x_2 \leq 3$ or $x_2 \geq 4$. With this we branch on the variable x_2 to create the following two subproblems:

Subproblem 1	Subproblem 2
max $\quad z = 5x_1 + 8x_2$	max $\quad z = 5x_1 + 8x_2$
s.t. $\; x_1 + x_2 \leq 6,$	s.t. $\; x_1 + x_2 \leq 6,$
$5x_1 + 9x_2 \leq 45,$	$5x_1 + 9x_2 \leq 45,$
$x_2 \leq 3,$	$x_2 \geq 4,$
$x_1, x_2 \geq 0.$	$x_1, x_2 \geq 0.$

On solving both the subproblems, we get the optimal solution for subproblem 1 as integers $x = (3, 3)$ with $z = 39$, and the optimal solution for subproblem 2 as $x = (1.8, 4)$ with $z = 41$.

In this case, we can fathom subproblem 1 because its solution has integers. The best integer solution found so far is stored as incumbent. The value of the incumbent is denoted by z^*. In this case, the first incumbent is $(3, 3)$, and $z^* = 39$. z^* is a lower bound for the optimal value of the integer programming problem : $z_{opt} \geq z^*$, or $z_{opt} \geq 39$.

We now branch the subproblem 2, on the variable x_1. The new subproblems 3 and 4 have restriction on x_1 as $x_1 \leq 1$ and $x_1 \geq 2$ respectively. The subproblem 3 has an optimal solution $x = (1, 4.444)$ with $z = 40.55$, and the subproblem 4 is infeasible, so it is fathomed. The upper bound for z_{opt} is updated as $39 \leq z_{opt} \leq 40.55$.

Now the subproblem 3 is branched on x_2. The new subproblems 5 and 6 have restriction on x_2 as $x_2 \leq 4$ and $x_2 \geq 5$ respectively. The subproblem 5 has optimal solution $x = (1, 4)$ with $z = 37$, and the subproblem 6 has just one point in its feasible region, which is its optimal solution : $x = (0, 5)$ with $z = 40$.

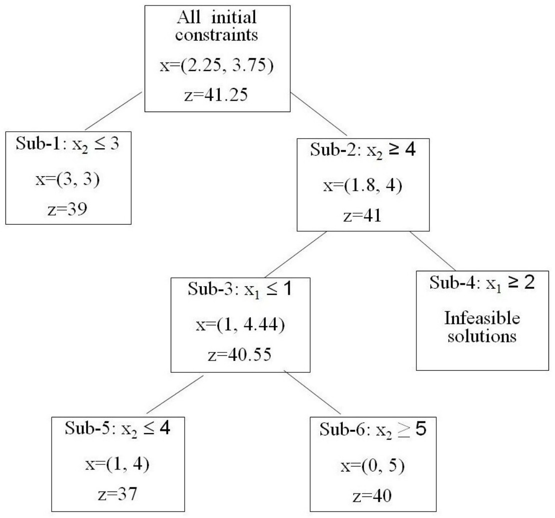

Figure 5.1: Final Solution Tree.

The subproblem 5 is fathomed because its optimal value $37 \leq 39 = z^*$. If a subproblem has integer optimal solution x^*, and its value is greater than z^*, then x^* replaces the current incumbent. Here, subproblem 6 has integer optimal solution, and its value $40 > 39 = z^*$. Thus, $(0, 5)$ is the new incumbent, and new $z^* = 40$. If there are no unfathomed subproblems left, then the current incumbent is an optimal solution for the given integer programming problem.

Thus, $(0, 5)$ is an optimal solution with optimal value 40.

Chapter 6

Stochastic Programming

Many optimization problems are described by uncertain parameters. There are different ways of incorporating this uncertainty. One of the approaches is stochastic programming, which assumes that the uncertain parameters are random variables with known probability distributions. This information is then used to transform the stochastic program into a so-called *deterministic equivalent* which might be a linear program, a nonlinear program or an integer program.

6.1 Stochastic Programming Problems

In stochastic programming, it is assumed that the set of all the K scenarios is a finite set $\omega = \{\omega_1, \ldots, \omega_K\}$ each corresponding to probabilities $\{p(\omega_1), \ldots, p(\omega_K)\}$, where $p(\omega_k) \geq 0$ for $k = 1, \ldots, K$, and satisfy $\sum_{k=1}^{K} p(\omega_k) = 1$.

Stochastic programming models can include variables corresponding to decisions that must be made *here-and-now* and cannot depend on the future observations/partial realizations of the random parameters, or on *wait-and-see* decisions that can be made after some of the random parameters are observed. Stochastic programming models which include both such types of variables are called as recourse models.

6.2 Two-Stage problems with recourse

A generic form of a two-stage stochastic linear program with recourse is of the following type :

$$\begin{gathered}\max\ z = c^T x + E[\max\ q(\omega)^T y(\omega)] \\ \text{s.t. } Ax = b, \\ T(\omega)x + W(\omega)y(\omega) = h(\omega), \\ x \geq 0, y(\omega) \geq 0.\end{gathered}$$

In the above formulation, the first-stage decisions are represented by vector x. These decisions are made *before* the random event ω is observed. The second-stage decisions are represented by the vector $y(\omega)$. These decisions are made *after* the random event ω has been observed. A and b define the deterministic constraints on the first-stage decisions x, whereas $T(\omega)$, $W(\omega)$, and $h(\omega)$ define the stochastic constraints linking the recourse decisions $y(\omega)$ to the first-stage decisions x. W is called the *recourse matrix*, which we assume here is fixed. Collecting the stochastic components of the problem, we obtain the vector $\xi^T(\omega) = (q(\omega)^T, h(\omega)^T, T_i(\omega))$, where $T_i(\omega)$ is i-th row of the technology matrix $T(\omega)$.

The second-stage or recourse problem for a particular realization ω of ξ is as follows :

$$\begin{aligned} \mathbb{Q}(x,\xi(\omega)) = & \max\ q(\omega)^T y(\omega) \\ \text{s.t. } W(\omega)y(\omega) &= h(\omega) - T(\omega)x, \\ y(\omega) &\geq 0. \end{aligned}$$

Let $\mathbb{Q}(x) = E_\xi[\mathbb{Q}(x,\xi(\omega))]$ denote the expected value of this optimum.

The expectation of the second-stage objective becomes:

$$E[\max\ q(\omega)^T y(\omega)] = \textstyle\sum_{k=1}^{K} p_k \max\ q(\omega_k)^T y(\omega_k).$$

Writing q_k for $q(\omega_k)$, p_k for $p(\omega_k)$, y_k for $y(\omega_k)$, T_k for $T(\omega_k)$ and h_k for $h(\omega_k)$, under the K scenario approach the two-stage stochastic linear programming problem takes the following form:

$$\begin{aligned} \max \quad z = c^T x + \textstyle\sum_{k=1}^{K}\ p_k \max\ q_k^T y_k \\ \text{s.t.} Ax \ = \ b, \\ T_k x \ + \ W y_k = h_k \text{ for } k = 1, \ldots, K, \\ x \geq 0, \\ y_k \geq 0 \text{ for } k = 1, \ldots, K. \end{aligned}$$

This is a deterministic linear programming problem called the *deterministic equivalent* of the original uncertain problem.

Chapter 7

L-Shaped Method

The general computation model in stochastic programming is to choose some initial decision that minimizes current costs plus the expected value of future recourse actions. With a finite number of second-stage realizations and all linear functions, we can always form the full deterministic equivalent linear program. With many realizations, this form of the problem becomes quite large. Methods that ignore the special structure of stochastic linear programs become quite inefficient. Taking advantage of structure is beneficial in stochastic programs and is the focus of much of the algorithmic work in this area. The method used most frequently is based on building an outer linearization of the recourse cost function and a solution of the first-stage problem plus this linearization. This cutting plane technique is called the *L-shaped method* in stochastic programming.

Consider the general formulation of the uncertain problem into its deterministic equivalent problem (D.E.P.) :

$$\begin{aligned} \min \quad & z = c^T x + \mathbb{Q}(x) \\ \text{s.t.} \quad & Ax = b, \\ & x \geq 0, \end{aligned}$$

where

$$\mathbb{Q}(x) = E_\xi \mathbb{Q}(x, \xi(\omega)),$$

and

$$\mathbb{Q}(x, \xi(\omega)) = min_y \{q(\omega)^T y | Wy = h(\omega) - T(\omega)x, y \geq 0\}.$$

7.1 L-Shaped Method

The basic idea of the L-shaped method is to approximate the nonlinear term in the objective of these problems.

The original problem

$$min \quad z = c^T x + \mathbb{Q}(x) \tag{7.1}$$

with constraints on x, is equivalent to solving

$$\begin{aligned} \min \quad & z = c^T x + \theta \\ & \mathbb{Q}(x) \leq \theta, \end{aligned}$$

where in both the problems, $\mathbb{Q}(x)$ is defined as :

$$\mathbb{Q}(x) = E_w \mathbb{Q}(x, \xi(w)),$$

and

$$\mathbb{Q}(x, \xi(w)) = min_y \{q(w)^T y | Wy = h(w) - T(w)x, y \geq 0\}.$$

7.2 Algorithm

The L-shape method adds two types of constraints :

(i) *Feasibility cuts*, and

(ii) *Optimality cuts.*

L-Shaped Algorithm

<u>*Step 0*</u> Set $r = s = \nu = 0$.

<u>*Step 1*</u> Set $\nu = \nu + 1$. Solve the linear program (3.2)-(3.4).

$$min \quad z = c^T x + \theta \tag{7.2}$$

$$\text{s.t.} \quad Ax = b,$$

$$D_l x \geq d_l, \quad l = 1, \ldots, r \tag{7.3}$$

$$E_l x + \theta \geq e_l, \quad l = 1, \ldots, s \tag{7.4}$$

$$x \geq 0, \qquad \theta \in \Re.$$

Let (x^ν, θ^ν) be an optimal solution. If no constraint (3.4) is present, θ^ν is set equal to $-\infty$ and is not considered in the computation of x^ν.

<u>*Step 2*</u> For $k = 1, \ldots, K$ solve the linear program

$$min \quad w' = e^T v^+ \ + \ e^T v^- \tag{7.5}$$

$$s.t. \quad Wy + Iv^+ - Iv^- = h_k - T_k x^v, \tag{7.6}$$

$$y \geq 0, \quad v^+ \geq 0, \quad v^- \geq 0,$$

where $e^T = (1, \ldots, 1)$, until, for some k, the optimal value $w' > 0$. In this case, let σ^ν be the associated simplex multipliers and define

$$D_{r+1} = (\sigma^\nu)^T T_k, \tag{7.7}$$

$$d_{r+1} = (\sigma^\nu)^T h_k, \tag{7.8}$$

to generate a constraint (called a *feasibility cut*) of type (3.3).
Set $r = r + 1$, add to the constraint set (3.3), and return to Step 1. If for all $k, w' = 0$, go to Step 3.

Step 3 For $k = 1, \ldots, K$ solve the linear program

$$min \quad w = q_k^T y \tag{7.9}$$

$$\text{s.t.} \quad Wy = h_k - T_k x^\nu,$$
$$y \geq 0.$$

Let π_k^ν be the simplex multipliers associated with the optimal solution of Problem k of type (3.9). Define

$$E_{s+1} = \sum_{k=1}^{K} p_k \cdot (\pi_k^\nu)^T T_k, \tag{7.10}$$

and

$$e_{s+1} = \sum_{k=1}^{K} p_k \cdot (\pi_k^\nu)^T h_k. \tag{7.11}$$

Let $w^\nu = e_{s+1} - E_{s+1} x^\nu$. If $\theta^\nu \geq w^\nu$, stop; x^ν is an optimal solution. Otherwise, set $s = s + 1$, add to the constraint set (3.4), and return to Step 1.

7.2.1 Example

The following is an example illustrating the optimality cuts on a stochastic programming problem.

Example 6 Let

$$\mathbb{Q}(x,\xi) = \{ \begin{array}{ll} \xi - x & \text{if } x \le \xi, \\ x - \xi & \text{if } x \ge \xi. \end{array}$$

Let ξ take the values 1, 2 and 4, each with probability 1/3. Assume also $c = 0$ and $0 \le x \le 10$. Figure 3.1 represents the functions $\mathbb{Q}(x,1)$, $\mathbb{Q}(x,2)$ and $\mathbb{Q}(x,4)$. Because the first-stage objective $c^T x$ is zero, $\mathbb{Q}(x)$ is also the function $z(x)$ to be minimized. There are no feasibility cuts as there is no $Ax = b$ constraint.

The initial given problem can be represented by the following graph :

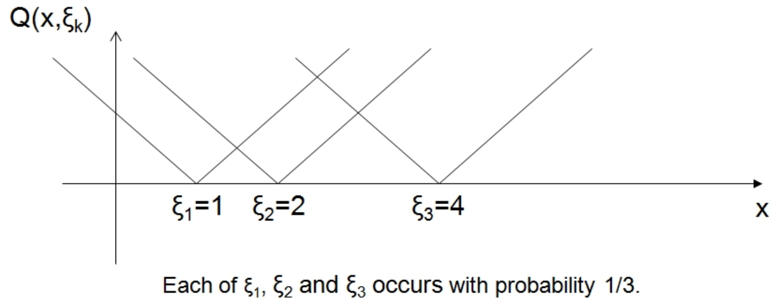

Figure 7.1: Initial Problem for $\mathbb{Q}(x,\xi)$.

The problem (P_1)

$$\min \mathbb{Q}(x)$$
$$\text{s.t. } 0 \le x \le 10,$$

is equivalent to solving (P_2)

$$\begin{aligned} &\min\ \theta \\ &\text{s.t. } 0 \le x \le 10, \quad \mathbb{Q}(x) \le \theta. \end{aligned}$$

The above problem has the following linear program as its subproblem, equivalent to step 3 of the algorithm, (P_{sp}) :

$$\begin{aligned} &\min\ y_1 \\ &\text{s.t. } y_1 - y_2 \qquad = \xi - x, \\ &\qquad y_1 \qquad - y_3 = -\xi + x, \\ &\qquad y_1, y_2, y_3 \ge 0. \end{aligned}$$

where,

$$W = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix};\ h_k = \begin{bmatrix} \xi \\ -\xi \end{bmatrix};\ T_k = \begin{bmatrix} 1 \\ -1 \end{bmatrix};$$

The sequence of iterations for the L-shaped method are as follows:

Iteration 1 (P_2) is solved where at initialization we assume $\theta^1 = -\infty$. We assume the starting point is $x^1 = 0$. As there is no feasibility issue, we go to Step 3 of the L-shaped algorithm, and solve (P_{sp}) :

The three simplex multipliers corresponding to each of the three values, 1,2 and 4, for ξ are: $\pi_1^1 = [1\ \ 0]$; $\pi_2^1 = [1\ \ 0]$; $\pi_3^1 = [1\ \ 0]$.
For each of $k = 1, 2, 3$, $p_k = 1/3$ and $T_k = [1\ \ -1]^T$.

Calculating :
$E_1 = \sum_{k=1}^{3} \frac{1}{3}[1\ \ 0] \begin{bmatrix} 1 \\ -1 \end{bmatrix} = 1$, and $e_1 = \sum_{k=1}^{3} \frac{1}{3}[1\ \ 0] \begin{bmatrix} \xi_k \\ -\xi_k \end{bmatrix} = 7/3$.

So, we have : $w^1 = e_1 - E_1 x^1 = 7/3$.
We have $\theta^1 < w^1$, so $x^1 = 0$ is not an optimal solution. Set $s = s + 1$, and

add the constraint : $\theta \geq e_1 - E_1 x$, or $\theta \geq 7/3 - x$ to the optimality cuts. Hence the first optimality cut is $\theta \geq 7/3 - x$.

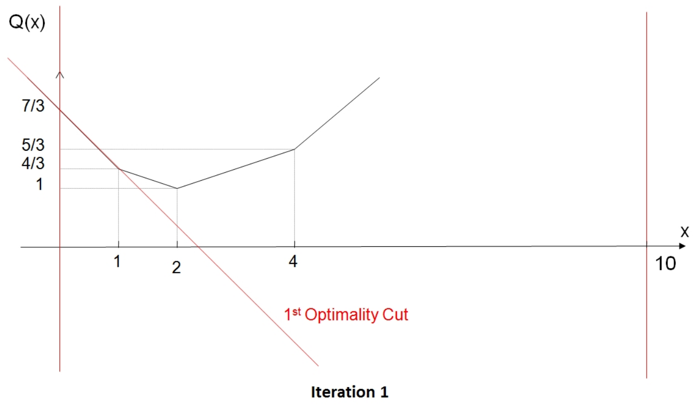

Figure 7.2: First Optimality cut.

Iteration 2 (P_2) is solved with the first optimality cut as an additional constraint and $x^2 = 10$ is obtained as an optimal solution, giving $\theta^2 = -23/3$. The simplex multipliers for this iteration are : $\pi_1^2 = [0 \;\; 1]$; $\pi_2^2 = [0 \;\; 1]$; $\pi_3^2 = [0 \;\; 1]$. $E_2 = -1$ and $e_2 = -7/3$, so $w^2 = 23/3$. Since $\theta^2 < w^2$, so we add second optimality cut $\theta \geq x - 7/3$.

Iteration 3 (P_2) is solved with the second optimality cut as an additional constraint and $x^3 = 7/3$ is obtained as an optimal solution, giving $\theta^3 = 0$. The simplex multipliers for this iteration are : $\pi_1^3 = [0 \;\; 1]$; $\pi_2^3 = [0 \;\; 1]$; $\pi_3^3 = [1 \;\; 0]$.

$E_3 = -1/3$ and $e_3 = 1/3$, so $w^3 = 10/9$. Again $\theta^3 < w^3$ so third optimality cut is added : $\theta \geq (x+1)/3$.

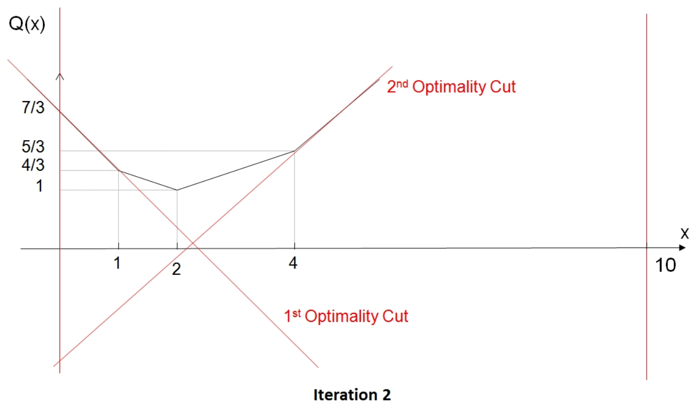

Figure 7.3: Second Optimality cut.

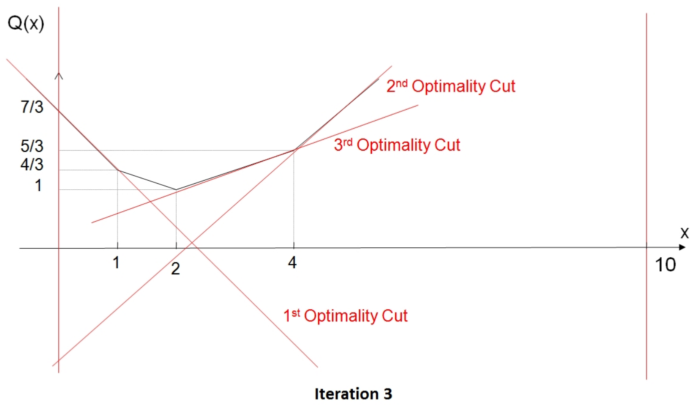

Figure 7.4: Third Optimality cut.

Iteration 4 (P_2) is solved with the third optimality cut as an additional constraint and $x^4 = 3/2$ is obtained as an optimal solution, giving $\theta^4 = 5/6$. $E_4 = 1/3$ and $e_4 = 5/3$, so $w^4 = 7/6$. Again $\theta^4 < w^4$ so fourth optimality cut is added : $\theta \geq (5 - x)/3$.

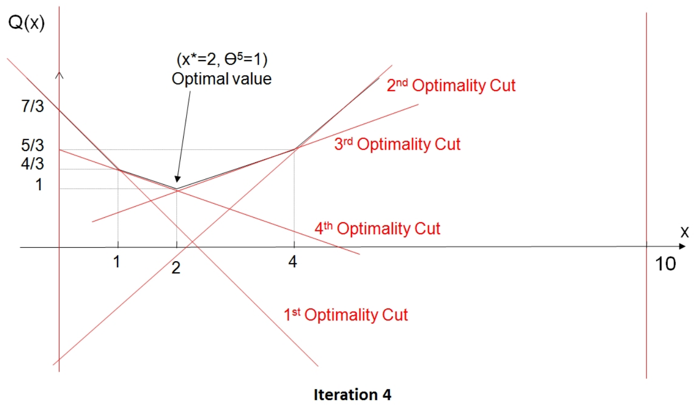

Figure 7.5: Fourth Optimality cut and the Optimal value.

Iteration 5 (P_2) is solved with the fourth optimality cut as an additional constraint and $x^5 = 2$ is obtained as an optimal solution, giving $\theta^5 = 1$. This step satisfies the optimality criteria $\theta^\nu \geq w^\nu$. Here $\theta^5 \geq w^5$ hence $x^* = 2$ is the optimal solution.

Chapter 8

Stochastic Integer Programs

8.1 Integer L-Shaped Method

A stochastic integer program is defined as :

$$min_{x\in X} \quad z = c^T x + E_\xi \; min\{q(w)^T y | Wy = h(w) - T(w)x, y \in Y\}$$
$$\text{s.t. } Ax = b,$$

where, the definitions of c, b, ξ, A, W, T and h are as before. However, X or Y might contain some integer constraints. Defining again the deterministic equivalent program of the form :

$$min_{x\in X} \quad z = c^T x + \mathbb{Q}(x)$$
$$\text{s.t. } Ax = b,$$

with $\mathbb{Q}(x)$ as the expected value of the second stage defined as in Section 2.2.

Let $\bar{\text{X}}$ be the polytope defined by the set of constraints in X other than those defining the first-stage variable type. Hence, for problems with integer first-stage variables, X= $\bar{\text{X}}$ $\cap Z_+^{n_1}$.

At step 2 of the Integer L-shaped algorithm, we consider the *current problem* :

$$
\begin{aligned}
&min \;\; z = c^T x + \theta \\
&\text{s.t.} \quad Ax \; = \; b, \\
&D_l x \;\geq\; d_l, \quad l = 1, \ldots, r, \\
&E_l x + \theta \geq e_l, \quad l = 1, \ldots, s, \\
&x \geq 0, \qquad \theta \in \Re.
\end{aligned}
$$

In the current problem the first-stage constraints include $Ax = b$ and $x \in \bar{\mathrm{X}}$, and the integer restrictions are relaxed in $x \geq 0$ and the restriction $x \in \bar{\mathrm{X}}$ is relaxed in feasibility cuts. Integer condition is recovered by the handling scheme which creates a number of nodes called as pendant as long as they are not examined.

Definition 8.1.1. A set of feasibility cuts is said to be valid at x if there exists a finite r such that

$$D_l x \geq d_l, l = 1, \ldots, r \text{ implies } x \in \bar{\mathrm{X}}.$$

Definition 8.1.2. A set of optimality cuts is said to be valid at $x \in X$ if there exists a finite s such that

$$(x, \theta) \text{ satisfies } E_l x + \theta \geq e_l, l = 1, \ldots, s$$

with $\theta \geq \mathbb{Q}(x)$.

8.1.1 Algorithm

Integer L-Shaped Algorithm

Step 0 Set Set $r = s = \nu = 0$, $\bar{\text{z}} = \infty$. The value of θ is set to $-\infty$ or to an appropriate lower bound and is ignored in the computation. A list is created that contains only a single pendant node corresponding to the initial subproblem.

Step 1 Select some pendant node in the list as the current problem; if none exists, stop.

Step 2 Set $\nu = \nu + 1$. Solve the current problem. If the current problem has no feasible solution, fathom the current node; go to Step 1. Otherwise, let (x^ν, θ^ν) be an optimal solution.

Step 3 Check for any relaxed constraint violation. If one exists, add one feasibility cut, set $r = r + 1$, and go to Step 2. If $c^T x^\nu + \theta^\nu > \bar{\text{z}}$, fathom the current problem and go to Step 1.

Step 4 Check for integrality restrictions. If a restriction is violated, create two new branches following the usual branch and bound procedure. Append the new nodes to the list of pendant nodes, and go to Step 1.

Step 5 Compute $\mathbb{Q}(x^\nu)$ and $z^\nu = c^T x^\nu + \mathbb{Q}(x^\nu)$. If $z^\nu < \bar{\text{z}}$, update $\bar{\text{z}} = z^\nu$.

Step 6 If $\theta^\nu \geq \mathbb{Q}(x^\nu)$, then fathom the current node and return to Step 1. Otherwise, impose one optimality cut, set $s = s + 1$, and return to Step 2.

8.2 Simple Integer Recourse

A two-stage stochastic program with simple integer recourse is as follows :

$$\begin{aligned} \min\ z = c^T x + E_\xi\{min(q^+)^T y^+ + (q^-)^T y^- | y^+ \geq \xi - Tx,& \\ y^- \geq Tx - \xi, y^+ \in Z_+^m, y^- \in Z_+^m\}& \\ \text{s.t. } Ax = b, x \in X,& \end{aligned}$$

where X defines non-negative integer decision variables and where $\xi = h$ because both T and q are known and fixed. We may replace the second-stage value function $\mathbb{Q}(x)$ by a separable sum over the various coordinates. Then $\mathbb{Q}(x)$ is separable in the components χ_i :

$$min \quad z = c^T x + \mathbb{Q}(x) \tag{8.1}$$

where $\mathbb{Q}(x) = \sum_{i=1}^m \Psi_i(\chi_i)$, with $\Psi_i(\chi_i) = E_{\xi_i}\Psi_i(\chi_i, \xi_i)$
and $\Psi_i(\chi_i, \xi_i) = min\{q_i^+ y_i^+ + q_i^- y_i^- | y_i^+ \geq \xi_i - \chi_i, y_i^- \geq \chi_i - \xi_i, y_i^+, y_i^- \in Z_+\}$.

Any error made in bidding χ_i versus ξ_i must be compensated by an integer in the second stage. Defining the expected shortage as

$$u_i(\chi_i) = E\lceil \xi_i - \chi_i \rceil^+ \tag{8.2}$$

and the expected surplus as

$$v_i(\chi_i) = E\lceil \chi_i - \xi_i \rceil^+, \tag{8.3}$$

where $\lceil x \rceil^+ = max\{\lceil x \rceil, 0\}$. It follows that $\Psi_i(\chi_i)$ is simply

$$\Psi_i(\chi_i) = q_i^+ u_i(\chi_i) + q_i^- v_i(\chi_i). \tag{8.4}$$

We assume $q_i^+ \geq 0, q_i^- \geq 0$.

The values of the expected shortage and expected surplus can be computed in finitely many steps. The function Ψ is generally not convex and discontinuous when ξ has a discrete distribution. But some form of convexity exists between function values evaluated in (not necessarily integer) points that are integer length apart.

Proposition 8.2.1. *Let $x^0 \in \Re, i, j \in Z$ with $j \leq i, x^1 = x^0 + i, x^\lambda = x^0 + j$. Then*

$$\Psi(x^\lambda) \leq \lambda\Psi(x^0) + (1-\lambda)\Psi(x^1) \tag{8.5}$$

with $\lambda = (i-j)/i$.

This means that we can draw a piecewise linear convex function through points that are integer length apart. Such a convex function is called a $\rho-$approximation rooted at x if it is drawn at points $x+\kappa$ or $x-\kappa$, κ being an integer.

8.2.1 Integer components of $\mathbb{Q}(x)$

When second-stage variables are integers, first-stage variables are expected to be integers leading to integer values for χ. It suffices then for T to have integer coefficients. By the definition of $\rho-$approximation rooted at an integer point, (4.1) is equivalent to

$$min\{c^T x + \sum_{i=1}^{m} \rho_i(\chi_i) | Ax = b, \chi = Tx, x \in X\}, \tag{8.6}$$

where T is such that $x \in X$ implies χ is integer, and ρ_i is a $\rho-$approximation of Ψ_i rooted at an integer point.

Because the objective in (4.6) is piecewise linear and convex, it can be solved by the L-shaped method. Using multicuts here will help in generating individual cut information for each subproblem that may require many cuts. Resulting to solve the following form of the sequence of current problems :

$$min_{x\in X,\theta\in\Re^m}\{c^T x+\sum_{i=1}^{m}\theta_i|Ax=b,\chi=Tx,E_{il}\chi_i+\theta_i\geq e_{il},i=1,.,m,\ l=1,.,s_i\}. \tag{8.7}$$

Above, the last set of constraints has optimality cuts. If χ_i^ν is a current iterate point with $\theta_i^\nu < \Psi_i(\chi_i^\nu)$, then an additional optimality cut is generated by dening

$$E_{ik} = \Psi_i(\chi_i^\nu) - \Psi_i(\chi_i^\nu + 1) \tag{8.8}$$

and

$$e_{ik} = (\chi_i^\nu + 1)\Psi_i(\chi_i^\nu) - \chi_i^\nu\Psi_i(\chi_i^\nu + 1). \tag{8.9}$$

The algorithm iteratively solves the current problem (4.7) and generates optimality cuts until an iterate point (χ^ν, θ^ν) is found such that $\theta_i^\nu = \Psi_i(\chi_i^\nu), i = 1, \ldots, m$. The algorithm is applicable for any type of random variables for which Ψ_is can be computed.

Chapter 9

Production Problem

Consider two products $i = 1, 2$, which can be produced by two machines $j = 1, 2$. Demand for both the goods follows a poission distribution with expectation 3. Production costs (in dollars) and times (in minutes) of the two products on the two machines are as follows :

Cost/Unit	Machine 1	Machine 2
Product 1	3	2
Product 2	4	5

Table 9.1: Production cost(in dollars) per unit for Machines.

Time/Unit	Machine 1	Machine 2	Finishing 1	Finishing 2
Product 1	20	25	4	7
Product 2	30	25	6	5

Table 9.2: Production and Finishing time(in minutes) per unit for Machines.

The total time for each machine is limited to 100 minutes. After machining, the products must be finished. Finishing time is a function of the machine used, with total available finishing time limited to 36 minutes. Production and demand correspond to an integer number of products. Product 1 sells at \$4 per unit. Product 2 sells at \$6 per unit. Unsold goods are lost. Defining x_{ij} = number of units of product i produced on machine j and $y_i(\xi)$ = amount of product i sold in state ξ. The problem takes the following form:

$$\begin{gathered}
\min\ 3x_{11} + 2x_{12} + 4x_{21} + 5x_{22} + E_\xi\{-4y_1(\xi) - 6y_2(\xi)\} \\
\text{s.t. } 20x_{11} + 30x_{21} \leq 100, \quad y_1(\xi) \leq \xi_1, \\
25x_{12} + 25x_{22} \leq 100, \quad y_2(\xi) \leq \xi_2, \\
4x_{11} + 7x_{12} + 6x_{21} + 5x_{22} \leq 36, \quad y_1(\xi) \leq x_{11} + x_{12}, \\
x_{ij} \geq 0 \ \text{ integer}, \ \ y_2(\xi) \leq x_{21} + x_{22}, \\
y_1(\xi), y_2(\xi) \geq 0 \ \text{ integer.}
\end{gathered}$$

Letting $y_i^+(\xi) = \xi_i - y_i(\xi)$, one obtains an equivalent formulation,

$$\begin{gathered}
\min\ 3x_{11} + 2x_{12} + 4x_{21} + 5x_{22} + E_\xi\{4y_1^+(\xi) + 6y_2^+(\xi)\} - 30 \\
\text{s.t. } 20x_{11} + 30x_{21} \leq 100, \quad y_1^+(\xi) \geq \xi_1 - \chi_1, \\
25x_{12} + 25x_{22} \leq 100, \quad y_2^+(\xi) \geq \xi_2 - \chi_2, \\
4x_{11} + 7x_{12} + 6x_{21} + 5x_{22} \leq 36, y_1^+(\xi), y_2^+(\xi) \geq 0 \text{ and integer,} \\
x_{11} + x_{12} = \chi_1, \\
x_{21} + x_{22} = \chi_2, \\
x_{ij} \geq 0 \text{ and integer.}
\end{gathered}$$

This representation puts the problem under the form of a simple recourse model with expected shortage only. Thus, $v_i(\chi_i) = 0$ for $i = 1, 2$.

On starting with the null solution, i.e., $x_{ij} = 0, \chi_i = 0, i, j = 1, 2$ with $\theta_i = -\infty, i = 1, 2$. Computing $u(0) = E\lceil\xi\rceil^+ = \mu^+ = 3$; hence

$\Psi_1(0) = 12,\ \Psi_2(0) = 18$, where we have dropped the constant, -30, from the objective for these computations.

To construct the first optimality cuts, we compute $u(1) = u(0) - 1 + F(0) = 2 + 0.05 = 2.05$. Thus, $E_{11} = 4(3 - 2.05) = 3.8, e_{11} = 4(1 * 3 - 0 * 2.05) = 12$. Thus, defining the optimality cut : $\theta_1 + 3.8\chi_1 \geq 12$.
As $\chi_2 = \chi_1, E_{21}$ and e_{21} are just 1.5 times E_{11} asnd e_{11}, respectively, yielding the optimality cut $\theta_2 + 5.7\chi_2 \geq 18$.

The current problem now becomes, with the two added optimality cuts :

$$\begin{gathered}
\min\ 3x_{11} + 2x_{12} + 4x_{21} + 5x_{22} - 30 + \theta_1 + \theta_2 \\
\text{s.t. } 20x_{11} + 30x_{21} \leq 100, \\
25x_{12} + 25x_{22} \leq 100, \\
4x_{11} + 7x_{12} + 6x_{21} + 5x_{22} \leq 36, \\
x_{11} + x_{12} = \chi_1, \\
x_{21} + x_{22} = \chi_2, \\
\theta_1 + 3.8\chi_1 \geq 12, \\
\theta_2 + 5.7\chi_2 \geq 18, \\
x_{ij} \geq 0, \text{ integer.}
\end{gathered}$$

On solving the above problem, (done using the matlab code $IP.m$ for integer programming), we obtain the solution : $x_{11} = 0, x_{12} = 4, x_{21} = 1, x_{22} = 0, \theta_1 = -3.2, \theta_2 = 12.3$. We now compute $u(4) = u(0) + \sum_{l=0}^{3}(F(l) - 1) = 0.31936$ and $\Psi_1(4) = 1.277 > \theta_1$. Thus a new optimality cut is needed for Ψ_1. Because $\Psi(5) = 0.5385$, the cut is $0.739\chi_1 + \theta_1 \geq 4.233$. We have $u(1) = 2.05$, so $\Psi_2(1) = 12.3 = \theta_2$, so no new cut is generated for $\Psi_2(.)$.

At the next iteration, with the extra optimality cut on θ_1, we obtain a new solution of the current problem as $x_{11} = 0, x_{12} = 2, x_{21} = 3, x_{22} = 0, \theta_1 = 4.4, \theta_2 = 0.9$. So, two new optimality cuts are added here :

$$2.312\chi_1 + \theta_1 \geq 9.623$$
$$2.117\chi_2 + \theta_2 \geq 10.383.$$

The next iteration gives $x_{11} = 0, x_{12} = 3, x_{21} = 2, x_{22} = 0, \theta_1 = 2.688, \theta_2 = 6.6$ as a solution of the current problem. Because $\Psi_2(2) = 7.5 > \theta_2$, a new cut is generated, i.e., $3.467\chi_2 + \theta_2 \geq 14.435$. The next iteration point is $x_{11} = 0, x_{12} = 3, x_{21} = 2, x_{22} = 0, \theta_1 = 2.688, \theta_2 = 7.5$, which is the optimal solution, as here $\theta_2 = \Psi_2(2)$, with the total objective value -5.812.

Chapter 10

Conclusion

We have implemented the simple integer recourse on stochastic programming problem, where $\mathbb{Q}(x)$ is separable into integer components, leading to integer solutions. The production problem is solved, which is calculating the minimum production cost for two machines with the integer constraints on the number of produts produced.

Bibliography

[1] J.E.Mitchell, Gerard Cornuejols, Reha Tuutuncu *Optimization Methods In Finance.* , 219-232, Summer 2005.

[2] J.R. Brige, F. Louveaux. *Introduction to Stochastic Programming.* Springer , 1997.

[3] D. Avis, http://cgm.cs.mcgill.ca/ avis/courses/567/notes/,Winter 2008.

[4] J. Linderoth. Lectures. *Stochastic Integer Programming*, $www.engr.wisc.edu/ie/faculty/linderoth_jeffrey.html$.

[5] V. Melkonian. Lectures. *Branch and Bound method*, $www.qpdf.info/ppt/1/branchandboundalgorithmforintegerprogram.html$.

[6] Stochastic Programming Community Home Page, www.stoprog.org/.

[7] L. A. Wolsey. *Integer Programming.* John Wiley, New York, 1998.

[8] Guan, Y., Ahmed, S., Nemhauser, G.L.. *Cutting Planes for Multistage Stochastic Integer Programs.* Operations Research(2009) 287-298

[9] N.S. Kambo, *Mathematical Programming Techniques.* East-West Press Ltd., 1991.