6.2 The Branch and Bound Method for Solving Pure Integer Programming Problem (Land-Doig-Dakin’s Algorithm)

The algorithm for solving IP problems is quite similar in the structure to the algorithm for BIP presented in the preceding section. The only thing we need to change is the choice of the branching variable. In a BIP problem, the binary variable is fixed at 0 and 1, respectively, for the two new subproblems. However, integer restricted variable in a IP problem could have a very large number of possible integer values, and it would be inefficient to create and analyze many subproblems by fixing the variable at its individual integer values. Therefore, what is done instead is to create two new subproblems by specifying two ranges of values for the variable.

**Branching Rule for IP** - Select one unterminated subproblem and branch on a variable $x_j$ whose optimal value $\bar{b}_j$ in the LP-relaxation is fractional as follows:

$$ x_j \leq \lfloor \bar{b}_j \rfloor, \quad x_j \geq \lceil \bar{b}_j \rceil + 1 $$

where $\lfloor \bar{b}_j \rfloor$ defines the greatest integer less than or equal to $\bar{b}_j$.

**Land-Doig-Dakin’s Algorithm**

The algorithm is summarized on page 475 in the textbook.

**Example 27** Example 15.2 on page 478-479 in the textbook.

6.3 The Branch and Bound Method for Solving Knapsack Problem

The knapsack problem is an integer programming problem with a single constraint. The problem may be written generally as

$$ \max \quad z = c_1x_1 + c_2x_2 + \cdots + c_nx_n $$

$$ \text{s.t.} \quad a_1x_1 + a_2x_2 + \cdots + a_nx_n \leq b $$

$$ x_j \geq 0, \quad \text{integer, } j = 1, 2, \ldots, n $$

where $c_j$ is the profit obtained if item $j$ is chosen, $b$ is the total amount of a resource, and $a_j$ is the amount of the resource used by item $j$.

The implementation of the B&B method to this problem becomes much easier due to that the LP-relaxation of each subproblem may be solved by inspection. To see this, observe that $\frac{c_j}{a_j}$ may be interpreted as the profit earned by item $j$ for each unit of the used resource. Thus the best item has the largest value of $\frac{c_j}{a_j}$ and the worst item has the smallest value of $\frac{c_j}{a_j}$. Therefore, to solve LP-relaxation of the subproblems, we need only to compute all the ratios $\frac{c_j}{a_j}$ and put the best item in the knapsack, then put the second-best item in the knapsack. Continue in this fashion until the best remaining item will overfill the knapsack. Then fill the knapsack with as much of this item as possible.
Example 28  Consider the following 0/1 knapsack problem:

\[
\begin{align*}
\text{max} & \quad z = 16x_1 + 22x_2 + 12x_3 + 8x_4 \\
\text{s.t.} & \quad 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \\
& \quad x_j \in \{0, 1\}, \quad j = 1, \ldots, 4.
\end{align*}
\]

The ratios \(\frac{c_j}{a_j}\) for Example 28 are listed in the following table.

<table>
<thead>
<tr>
<th>Item</th>
<th>(\frac{c_j}{a_j})</th>
<th>Ranking (1=best, 4=worst)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item 1</td>
<td>(\frac{4}{5})</td>
<td>1</td>
</tr>
<tr>
<td>Item 2</td>
<td>(\frac{22}{7})</td>
<td>2</td>
</tr>
<tr>
<td>Item 3</td>
<td>(\frac{12}{4})</td>
<td>3</td>
</tr>
<tr>
<td>Item 4</td>
<td>(\frac{8}{3})</td>
<td>4</td>
</tr>
</tbody>
</table>

To solve the LP-relaxation of the above problem, we first set \(x_1 = 1\) and then solved resulting knapsack problem. After setting \(x_1 = 1\), \(14 - 5 = 9\) units of the resource remain. Next we include the second best item (item 2) in the knapsack by setting \(x_2 = 1\), and then \(9 - 7 = 2\) units of the resource remain. Now the best remaining item is item 3, and we will fill the knapsack with as much of item 3 as we can. Since there are only 2 units of the resource left over, we set \(x_3 = \frac{2}{4} = \frac{1}{2}\). Thus, an optimal solution to the LP-relaxation of the problem in Example 25 is \(x_1 = x_2 = 1, x_3 = \frac{1}{2}, x_4 = 0, z = 44\). The LP-relaxation optimal solution is fractional, so we should branch the original problem into two subproblems by setting \(x_3 = 1\) and \(x_3 = 0\). Try to continue to solve this problem as an exercise, and check your answer with the following B&B tree, where the depth first rule is used for selecting the subproblems.

Example 29  Example 15.4 on page 483-484 in the text book.
Summary of the course

1. Minimum Cost Network Flow Problem
   - Mathematical formulation: anslutningsmatris (node-arc incidence matrix)
   - Integrality property (heltalsegenskap) of optimal flows
   - Modeling

2. Special Cases of the Minimum Cost Network Flow Problem
   - Transportation problem, Assignment problem (Tillordningsproblem), Shortest path problem (Billigaste väg problem), Maximum flow problem.
   - Mathematical formulation
   - Modeling
   - The algorithm for finding the Shortest paths (Dijkstra’s algorithm and Ford’s algorithm), Bellmans equations for the Shortest path problem.
   - Variations of the Shortest path problem: Longest path problem (Dyraste väg problem), the path with the maximum capacity (Väg med maximal Kapacitet).

3. Minimum Spanning Tree (Billigaste Uppspännande träd) Problem and the Algorithm (Kruskal’s Algorithm, Prim’s Algorithm).

4. The Network Simplex Method
   - Determination of the basic feasible solution
   - The simplex algorithm
   - Sensitivity analysis (känslighetsanalys)

5. Formulating Integer Programming Problems
   - Read lecture notes 4 (Föreläsning 4) and 13.1-13.6, 13.10 in the text book

6. The Solution Methods for Integer Programming Problems
   - Optimality and relaxation: the relation between the optimal objective value of the relaxed problem and the original problem (lower bound, upper bound)
   - LP-relaxation
   - Lagrangian relaxation, Lagrangian duality, explicit expression of the dual function
   - Valid inequality (Giltiga olikheter)
   - Gomory’s cutting plane method (Plansnittningsmetod)
   - Covering (övertäckning), minimum covering (minimal övertäckning), valid inequality for 0/1 knapsack problem
   - Branch and bound method (Trädsökning): Land-Doig-Dakin’s algorithm
   - Branch and bound method for binary integer programming problem
   - Branch and bound method for mixed integer programming problem
   - Branch and Bound method for knapsack (kappsäcksproblem) problem and binary knapsack problem
   - You Should be able to solve 2-variable LP problem by graphical method (you have learnt this method in the course “Optimeringslära”)

The examination will contain 5 main questions, each possibly involving two or more sub-questions. The questions will be given in Swedish.