Assignment No : B3

Roll no : 4351

1 Title

0-1 Knapsack Problem

2 Problem Statement

Write a program to implement 0-1 Knapsack using Branch and Bound Approach

3 Learning Objectives

1. To learn the concept of branch and bound approach.
2. To study the design and implementation of branch and bound 0-1 knapsack algorithm.
3. To implement the given algorithm as efficiently as possible.

4 Learning Outcome

1. Branch and Bound Approach was learnt successfully
2. 2. Implementation of 0-1 Knapsack was successfully designed and completed using the required programming language

5 Theory

Branch and Bound Approach

Like backtracking, branch-and-bound is a technique for exploring an implicit directed graph. Consider a graph, which is usually acyclic or even a tree. We are looking for the optimal solution to some problem. At each node we calculate a bound on the possible value of any solutions that might happen to be farther on in the graph. If the bound shows that any such solution must necessarily be worse than the best solution we have found so far, then we do not need to go on exploring this part of the graph. In the simplest version, calculation of these bounds is combined with a breadth first or a depth-first search, and serves only, to prune certain branches of a tree or to close certain paths in a graph. More often, however, the calculated bound is used not only to close off certain paths, but also to choose which of the open paths looks the most promising, so that it can be explored first. In general terms we may say that a depth-first search finishes exploring nodes in inverse order of their creation, using a stack to hold those nodes that have been generated but not yet explored fully ; a breadth-first search finishes exploring nodes in the order of their creation, using this time a queue to hold those that have been generated but not yet explored. Branch-and-bound uses auxiliary computations to decide at each instant which node should be explored next, and a priority list to hold those nodes that have been generated but not yet explored.

Knapsack Problem

There are many different knapsack problems. The first and classical one is the binary knapsack problem. It has the following story. A tourist is planning a tour in the mountains. He has a lot of objects which may be useful during the tour. For example ice pick and can opener can be among the objects. We suppose that the following conditions are satisfied.

- Each object has a positive value and a positive weight.
- The objects are independent from each other.

- The knapsack of the tourist is strong and large enough to contain all possible objects.
- The strength of the tourist makes possible to bring only a limited total weight.
- But within this weight limit the tourist want to achieve the maximal total value.

6 Mathematical Model

Let S be the solution perspective of the system such that
S={ s, e , X, Y, Fme , DD, NDD ,Ffriend $\mid \phi s$ }
where
DD = Deterministic Data
NDD = Non - Deterministic Data = User Input

ϕs = Constraints on System S

s is start state

e is end state=Weights Distributed successfully

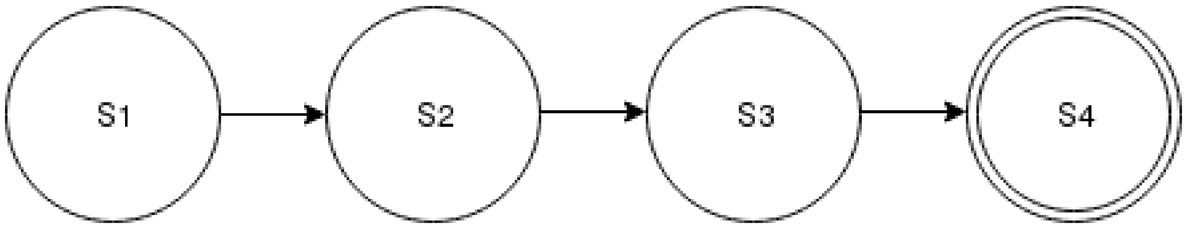
X=set of input parameters
X = X1,X2,X3
where, X1=Number of objects
X2 = weight
X3 = Value

Y=set of output parameters
Y = Y1
where Y1 = Knapsack Distribution according to values
Fme is the set of main functions
Fme = f1, f2, f3
f1 = find branch and bound
f2 = find profits according to the algorithm

Success Case = Right Solution

Failure Case = Wrong solution

7 State Diagram



s1 is the start state
s2 is the input state
s3 is the operational state
s4 is the final state

8 Conclusion

The concept of branch and bound strategy is studied and 0-1 knapsack algorithm is implemented.