CS472 Module 9 Part A - Branch and Bound - 0-1 Knapsack and the TSP

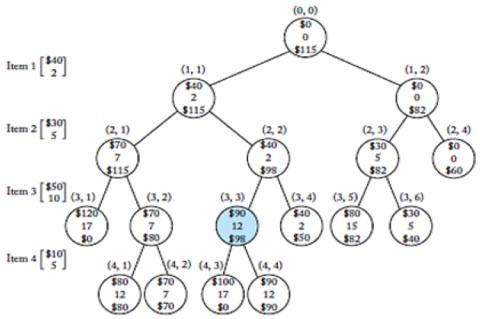
Athens State University

March 28, 2016

0-1 Knapsack? Again?

- Start by letting `weight` be the total weight and `profit` be the total profit of items included up to a node
- Keep track of the `totweight` and `bound`
- Greedily grab items adding weight to `totweight` and profit to `bound`
- Until an item's weight would bring `totweight` about the capacity of the sack
- Grab the fraction of the item allowed by available and add profit of the fraction to `bound`
- A node is non-promising if `bound` is $\leq$ `maxprofit`, the value of the best solution up to this point
- A node is also non-promising if `weight` $\geq$ capacity of the sack

Example



Can we do better?

- After visiting all of the children of a given node, expand the one with the best bound
- Determine promising unexpanded node with greatest bound
- Order determined by best bound rather than pre-determined
- Implement using a priority queue rather than simple queue (what is the effect?)

Back to our old friend, the TSP

- Goal: find an optimal tour
  - Start at a given node
  - Visits every node exactly once
  - Return to the starting node
- Optimal tour will be the one with the minimal total distance traveled

Bound for the TSP

- Determine lower bound on the length of any tour obtainable by expanding beyond given node
- Length of an edge taken when leaving a vertex $\geq$ length of shortest edge leaving that vertex
- Lower bound on any tour of n nodes is

$$\sum_{i \leq j \leq n, j \neq 1} \min W[i, j]$$

- The sum the minimum entry in each row of the adjacency matrix
- Expand node i representing a sub-path in the state space tree

Bound for the TSP

- The bound is cost of traveling the sub-path to arrive at node i plus the cost of minimum sub-path to travel from the last node in the path back to the start
- A node is non-promising of bound $\geq$ minlength