中山大學
SUN YAT-SEN UNIVERSITY

Lecture 10
Branch-and-Bound

Algorithm Design and Analysis

zhangzizhen@gmail.com

Branch-and-bound

- The branch-and-bound design strategy is very similar to **backtracking** in that a state space tree is used to solve a problem.
- The differences are that the branch-and-bound method 1) does not limit us to any particular way of traversing the tree, and 2) is used for optimization problems.
- A branch-and-bound algorithm computes a number (**bound**) at a node to determine whether the node is promising.

Branch-and-bound

- The number is a bound on the value of the solution that could be obtained by expanding beyond the node.
- If that bound is no better than the value of the best solution found so far, the node is **nonpromising**. Otherwise, it is **promising**.
- Besides using the bound to determine whether a node is promising, we can compare the bounds of promising nodes and visit the children of the one with the best bound.
- This approach is called **best-first search with branch-and-bound pruning.**

Branch-and-bound

- An enhancement of backtracking
- Applicable to optimization problems
- Uses a bound for the value of the objective function for each node (partial solution) so as to:
  - guide the search through state-space
  - rule out certain branches as “unpromising”
- Many Branch-and-bound methods are adopted using the Depth-First Search (DFS) manner, since it generally consumes little memories.

Breadth-first Search

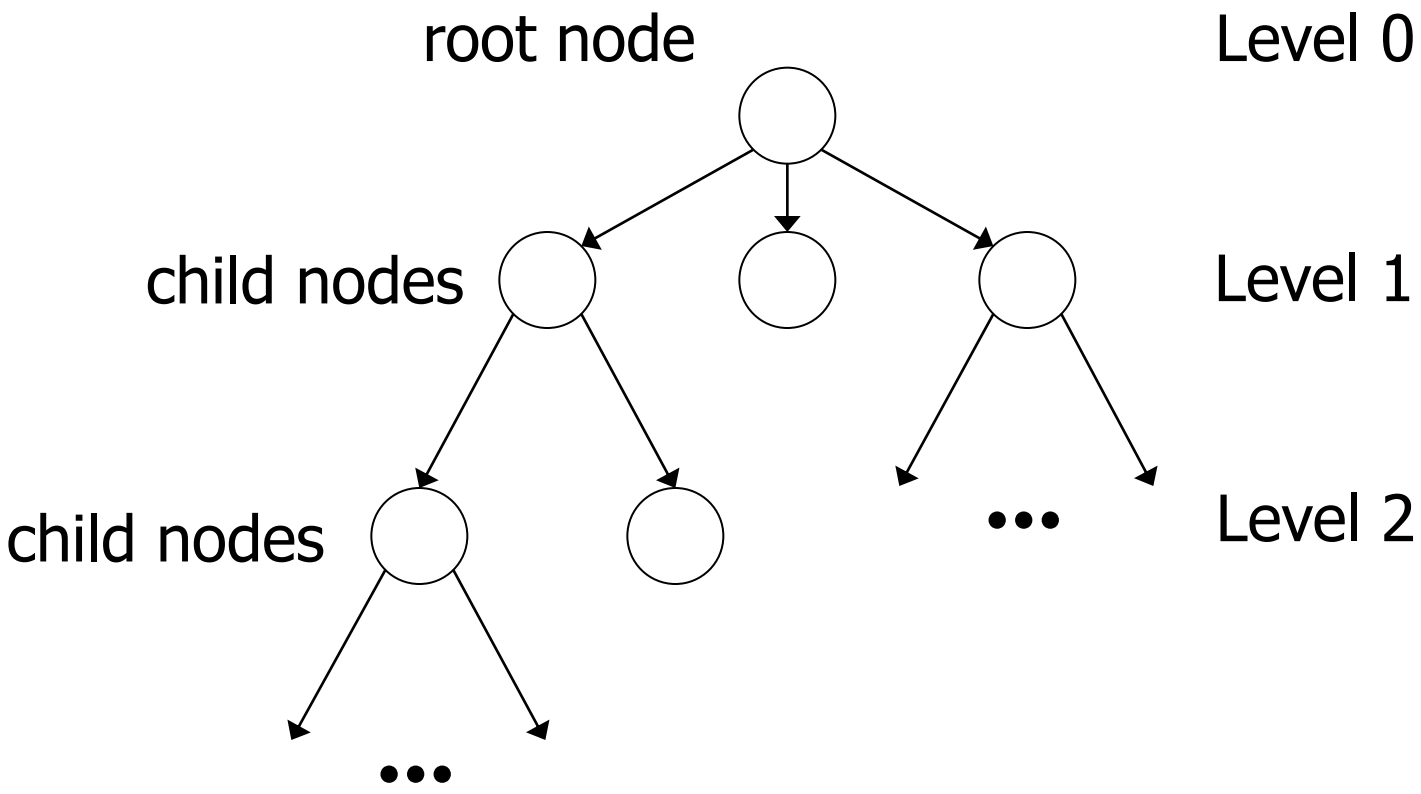
- The Branch-and-bound can also be applied within the Breadth-First Search (BFS) framework.
- We can implement the BFS using a queue.
- All child nodes are placed in the queue for later processing if they are promising.
- Consider the **minimization** problems.
- Calculate the value for each node that represents the minimum possible value if we pick that node.
- If the minimum possible value is greater than the global best value, don’t expand the branch.

Best-first Search

- The breadth-first search strategy has no advantage over a depth-first search (backtracking).
- However, we can improve our search by using our bound to do more than just determine whether a node is promising.
- Best-first search expands the node with the best bounds next. (A* ?)
- How would you implement a best-first search?
  - Depth-first is a stack
  - Breadth-first is a queue
  - Best-first is a ???

Enumeration in a search tree

- Each node is a partial solution, i.e. a part of the solution space



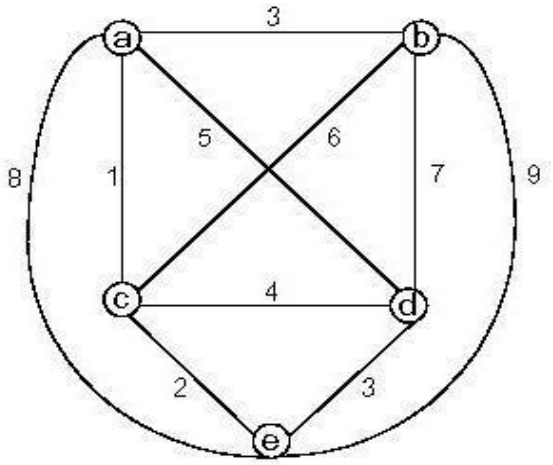
Bounding

- A bound on a node is a guarantee that any solution obtained from expanding the node will be:
  - Greater than some number (lower bound)
  - Or less than some number (upper bound)
- If we are looking for a maximal optimal (knapsack problem), then we need an upper bound.
- If we are looking for a minimal optimal (traveling salesman problem), then we need a lower bound.

Bounding

- We prune (via bounding) when:
  - (nodeBound >= currentBestSolutionCost) for minimization problem
  - (nodeBound <= currentBestSolutionCost) for maximization problem
- We want to find a globally good solution quickly.
- We want to find a good bound for each node.
- Initially, a global solution can be obtained by a greedy or an efficient heuristic.
- A bound can be obtained by relaxing some constraints of the subproblem related to the node.

Traveling Salesman Problem



- Generate permutations

TSP Bound

- Can apply branch & bound if we come up with a reasonable lower bound on tour lengths.
- Simple lower bound = finding smallest element in the intercity distance matrix D and multiplying it by number of cities *n*
- Another bound: $\frac{1}{2}\sum_{v \in V}$ (Sum of the costs of the two tour edges adjacent to v)
- 1-tree bound: a tour consists of a special *spanning tree* (namely a path) on the remaining *N*-1 nodes plus two edges connecting node 1 to this spanning tree.

0-1 Knapsack Problem

- Generate subsets
- Fractional knapsack bound

Summary: Branch-and-bound

- If an optimal solution is found to a subproblem, it is a feasible solution to the full problem, but not necessarily globally optimal.
- Since it is feasible, it can be used to prune the rest of the tree.
- The search proceeds until all nodes have been solved or pruned, or until some specified threshold is meet between the best solution found and the lower bounds on all unsolved subproblems.

Thank you!