中山大學
SUN YAT-SEN UNIVERSITY

Lecture 16
Recap

Algorithm Design

zhangzizhen@gmail.com

QQ group: 117282780

Basic Knowledge

- Characteristics of algorithms
- Scientific notation for numbers
- Numerical errors
- Bitwise operations
- Asymptotic complexities
- Order of growth
- Best, average, worst case analysis
- Mathematical induction
- Recursion
- …

Data Structures

- Stacks
- Queues
- Lists: Arrays and Linked lists
- Heap: Priority Queue
- Trees: Binary search trees, Tree traverse, Tree representation
- Hashing
- Disjoint-sets

Greedy

- Definition: A *greedy algorithm* is an algorithm in which at each stage a locally optimal choice is made.
- Characteristics:

  1. Greedy-choice property: A global optimum can be arrived at by selecting a local optimum.

  2. Optimal substructure: An optimal solution to the problem contains an optimal solution to subproblems.
- Greedy algorithms are usually extremely efficient, but they can only be applied to a small number of problems.

Greedy

- Examples:
  - Activity selection problem
  - Fractional Knapsack problem
  - Spanning trees
  - Huffman coding
  - Change-making problem (找零问题)
  - Biggest number: 设有n个正整数，将它们连接成一排，组成一个最大的多位整数。例如：n=3时，3个整数13，312，343，连成的最大整数为34331213。又如：n=4时，4个整数7，13，4，246，连成的最大整数为7424613。输入N个数，输出：连成的多位数

Divide-and-conquer

- A divide-and-conquer algorithm works by recursively breaking down a problem into two or more sub-problems of the same (or related) type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem.

- Example:
  - Mergesort / Quicksort
  - Strassen's Matrix Multiplication
  - Hanoi Tower

Dynamic Programming

- Dynamic Programming (DP) is a method for solving complex problems by breaking them down into simpler subproblems. It is applicable to problems exhibiting the properties of overlapping subproblems (子问题重叠) and optimal substructure (最优子结构).
- A problem is said to have overlapping subproblems if the problem can be broken down into subproblems which are reused several times or a recursive algorithm for the problem solves the same subproblem over and over rather than always generating new subproblems.
- A problem is said to have optimal substructure if an optimal solution can be constructed efficiently from optimal solutions of its subproblems.

Dynamic Programming

- You need to know the following concepts:
  - State (状态): A way to describe a situation, a sub-solution for the problem.
  - Top-down approach
  - Bottom-up approach
  - Memoization (记忆化)
  - Recurrence formulas / State transition equation (状态转移方程)
  - Traceback (for obtaining concrete solutions)

Dynamic Programming

- Examples:
  - Street Walking：一个城市的街道布局如下，从最左下方走到最右上方，每次只能往上或往右走，一共有多少种走法?
  - Weighted Street Walking
  - Longest Common Subsequence: O(n^2)
  - Longest Non-Decreasing Subsequence: O(n^2) -> O(n log n)
  - 0-1 Knapsack Problem: O(nW)
  - Matrix-Chain Multiplication: O(n^3)
  - Traveling Salesman Problem O(n!) -> O(n^2*2^n)

Search

- Backtracking: recursively explore each node.
- Branch-and-bound: It is very similar to backtracking. It computes a number (**bound**) at a node to determine whether the node is promising. If that bound is no better than the value of the best solution found so far, the node is **non-promising**. Otherwise, it is **promising**.
- Branch-and-bound methods are adopted using the Depth-First Search (DFS) manner, the Breadth-First Search (BFS) manner, and the Best-First search manner.
- Use bounds to do pruning.
- A bound can be obtained by relaxing some constraints of the subproblem related to the node.

Search in Graphs

- Uninformed Search
  - Uninformed means that we only know the goal and the ***succs()*** function.
  - Search strategies: BFS, UCS, DFS, IDS, Bi-directional search.

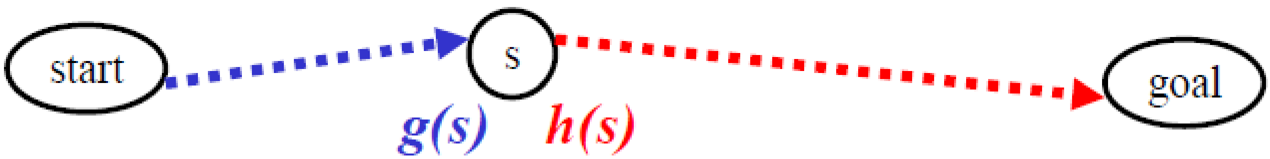
b: branching factor (assume finite)   d: goal depth   m: graph depth

	Complete	optimal	time	space
Breadth-first search	Y	Y, if [1]	$O(b^d)$	$O(b^d)$
Uniform-cost search	Y	Y	$O(b^{C^*/\varepsilon})$	$O(b^{C^*/\varepsilon})$
Depth-first search	N	N	$O(b^m)$	$O(bm)$
Iterative deepening	Y	Y, if [1]	$O(b^d)$	$O(bd)$
Bidirectional search	Y	Y, if [1]	$O(b^{d/2})$	$O(b^{d/2})$

1. edge cost constant, or positive non-decreasing in depth

Search in Graphs

- Informed Search
  - Informed means that we also know a heuristic ***h(s)*** of the cost from node ***s*** to goal.



  - Search strategies: A*, IDA*.
  - For the heuristic function *h()*, if ***h(s)<=h*(s),*** where ***h*(s)*** is the true cost from node s to the goal, then such heuristic function ***h()*** is called ***admissible.***

Search

- Examples:
  - Subset Tree Enumeration
  - Permutation Tree Enumeration
  - Sudoku
  - Traveling Salesman Problem
  - 8-Puzzle Problem
  - Arithmetic Calculation Problem
  - Graph Coloring Problem (图染色)
  - Maximum Clique Problem (最大团)
  - Many Path Finding Problems

Graph Algorithms

- Graph representations: adjacency matrix, adjacency list
- Graph traverse: DFS, BFS
- Topological sorting for DAGs: It is a linear ordering of nodes in which each node comes before all nodes to which it has outbound edges.
- Single source shortest path algorithms: Dijkstra algorithm (O(n^2) or O(E+VlogV)), Bellman-ford algorithm
- All-pairs shortest path algorithms: Floyd-Warshall algorithm
- System of difference constraints
- Minimum spanning tree algorithms: Prim’s algorithm, Kruskal’s algorithm

Optimization Algorithms

- Approximation algorithms: providing provable solution quality and provable run-time bounds.
- Heuristic algorithms: the art of discovering new strategies (rules) to solve problems.
- Meta-heuristic algorithms: upper level general methodologies that can be used as guiding strategies in designing underlying heuristics to solve specific optimization problems
- Components in SA, TS, GA

Network flows

- Maximum flow
  - Ford-Fulkerson Algorithm
- Minimum cut
- Maximum matching in bipartite graph

Thank you!