

CSE 361: Complexity of Sequential and Parallel Algorithms

Exam II; Help Sheet

GREEDY ALGORITHMS. This technique is used when we are interested in finding a subset of n given objects that satisfies a set of constraints and optimizes a given objective function. The general idea is to start with the empty set; select the next object O to be examined using a selection criterion; if the inclusion of O into the solution S will still keep it feasible we add O into S , otherwise we discard O ; proceed in a similar fashion until all the objects have been examined; and finally output the solution S .

We were able to solve the fractional knapsack problem in $O(n \log n)$ time using the greedy approach. The idea is to process the objects in nonincreasing order of their profit densities.

We also showed that the minimum weight spanning tree problem can be solved in $O((|V| + |E|) \log |V|)$ time on any weighted undirected graph $G(V, E)$ employing the greedy technique. Prim's algorithm has only one tree at any time. It looks at all the outgoing edges from the tree and includes the edge with the minimum weight. Kruskal's algorithm starts with a forest of n trees and inserts one edge at a time into the forest (if the edge does not cause a cycle). The edges are sorted in nondecreasing order of the edge weights to begin with.

Dijkstra's algorithm for the single source shortest path problem runs in time $O(|E| \log |E|)$ time. This algorithm assumes that the input graph does not have any edges with negative weights.

DYNAMIC PROGRAMMING. Dynamic programming applies to problems for which the solutions can be thought of sequences of decisions. In addition, the principle of optimality should hold for the problem.

The general solution technique here typically involves the following steps: 1) define a suitable function such that the outputs of interest are specific values of this function; 2) write a recurrence relation for this function; and 3) solve the recurrence relation to get the values of interest – the base cases for the function are usually the inputs.

In the all-pairs shortest paths problem, the input is a directed graph $G(V, E)$. The goal is to find the shortest path from the node i to node j for every pair of nodes i and j in V . We define the function $A^k(i, j)$ to be the shortest path length from i to j from among all paths (from i to j) whose intermediate nodes are $\leq k$. We are interested in the values $A^n(i, j)$ (for every i and j in V), where $n = |V|$. A recurrence relation for $A^k(i, j)$ can be written as follows:

$$A^k(i, j) = \min\{A^{k-1}(i, j), A^{k-1}(i, k) + A^{k-1}(k, j)\}.$$

We start with A^0 and compute A^1, A^2, \dots, A^n . The total run time is $O(n^3)$.

Bellman and Ford's algorithm for the single source shortest path problem runs in time $O(|V| |E|)$ and applies to graphs with general edge weights (as long as the graph does not have any negative cycles). The idea here is to define a function $dist^l[u]$ for every node u in $V - \{s\}$. $dist^l[u]$ is the length of the shortest path from s to u from among all paths from s to u that have at most l edges. We are interested in computing $dist^{n-1}[u]$ for every node u .

String editing takes as input two strings $X = x_1x_2 \cdots x_n$ and $Y = y_1y_2 \cdots y_m$. The problem is to transform X into Y so that the edit cost is minimum. Allowed operations are INSERT, DELETE, and CHANGE. Here one defines $cost(i, j)$ to be the minimum cost needed to transform $x_1x_2 \cdots x_i$ into $y_1y_2 \cdots y_j$. We can find the optimal edit cost in $O(mn)$ time.

For the zero-one knapsack problem, we define $f_i(y)$ to be the optimal profit obtainable from the objects 1 through i when the capacity constraint is y . A recurrence relation for $f_i(y)$ takes the form:

$$f_i(y) = \max\{f_{i-1}(y), f_{i-1}(y - w_i) + p_i\}.$$

When the weights are integers, the above recurrence relation can be used to solve the problem in $O(mn)$ time. When the weights are general we solve the zero-one knapsack problem in $O(2^n)$ time constructing the sets S^1, S^2, \dots, S^n .

BASIC GRAPH ALGORITHMS. There are many ways to conduct tree traversals (such as In-Order, Pre-Order, and Post-Order). These algorithms take linear time. Depth First Search (DFS) and Breadth First Search (BFS) can be used to conduct generic graph searches. These algorithms take $O(|V| + |E|)$ time.