# CSE 5500: Algorithms

## Exam II; Help Sheet

**Algebraic Problems**. A degree-$n$ polynomial can be evaluated at a given point in $O(n)$ time. Lagrangian interpolation algorithm runs in $O(n^3)$ time whereas Newton's interpolation algorithm takes $O(n^2)$ time.

Two degree-$n$ polynomials can be multiplied in $O(n \log n)$ time. This algorithm is based on Fast Fourier Transform (FFT). A degree-$n$ polynomial can be evaluated at $n$ given arbitrary points in $O(n \log^2 n)$ time. Also, interpolation of a polynomial presented in value form at $n$ arbitrary points can be done in $O(n \log^3 n)$ time.

We also discussed three applications for FFT, namely, a problem in communication complexity, secret sharing, and similarity measurement between two strings.

**Graph Algorithms.** Depth first search (DFS) and breadth first search (BFS) are some basic graph search algorithms. These algorithms take $O(|V| + |E|)$ time on any input graph $G(V, E)$. These algorithms have numerous applications. For example, we can find the connected components of a given graph using DFS or BFS.

We 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.

**Dynamic Programming.** The general dynamic programming solution technique 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 $A^1, A^2, \ldots, A^n$. The total run time is $O(n^3)$.

String editing takes as input two strings $X = x_1 x_2 \cdots x_n$ and $Y = y_1 y_2 \cdots y_m$. 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 $x_1 x_2 \cdots x_i$ into $y_1 y_2 \cdots y_j$. We can find the optimal edit cost in $O(mn)$ time.

**Maxflow Algorithms.** A flow network is a directed graph $G(V, E)$ where each edge has a capacity, one of the nodes is the source and another node is the sink. The problem is to identify the maximum flow that can be sent from the source to the sink. We described the Ford-Fulkerson algorithm. We also proved the maxflow-mincut theorem. When the capacities are integers and if the maximum flow value is $f*$, then we proved that the run time of the Ford-Fulkerson algorithm is $O(f * (|V| + |E|))$.