CS 341 — Lecture 11

Bartosz Antczak *Instructor: Bin Ma* *October 17, 2017*

Final Note on DP

Divide and Conquer involves applying the same algorithm on a smaller size of input that **don't rely** on one another; whereas, dynamic programming also builds on smaller sub-problems but the main difference is that they **do rely** on the smaller sub-problems.

11.1 Graph Algorithms

This week we'll start learning about graph algorithms.

11.1.1 Terminology

A **graph** is a set of vertices and edges (flashback to MATH 239). We use notation $G = (V, E)$. We define an edge as $e = (u, v)$. This means that vertices u, v are *adjacent* and e, u or e, v are *incident*. A graph can also be directed.
A graph is **connected** if there exists a path between every vertex u, v.
An **acyclic** graph is a graph without a cycle. Observe that an acyclic undirected graph is a *tree*.

11.1.2 Storing a Graph in a Computer

- **Adjacency Matrix:**

$$A[i,j] = \begin{cases} 1 & \text{if } i,j \in E \\ 0 & \text{Otherwise} \end{cases}$$

  . **Space complexity:** $O(|V|^2)$

- **Adjacency List:** the index of the array represents the vertex number, and the value at every index is the list of adjacent vertices. **Space complexity:** $O(|V| + |E|)$

11.1.3 Graph Traversal

Graph traversal is the process of visiting every vertex in certain order.

Breath-First Search

Split the graph into k layers where a vertex at $i \in [1, k]$ takes i steps to access it. Given an input of a graph $G = (V, E)$ and a starting index $s \in V$, the breath-first search algorithm is (*on next page*):

The Algorithm

$Q \leftarrow$ empty queue
visited$[v] \leftarrow$ false for every $v \in V$
enque(Q, s)
visited$[s] \leftarrow$ true
while Q is not empty **do**
  $u \leftarrow$ dequeue(Q)
  for Every v adjacent to u **do**
    if !visited$[v]$ **then**
      enque(Q, v)
      visited$[v] \leftarrow$ true
      parent$[v] \leftarrow u$
      level$[v] \leftarrow$ level$[u] + 1$

Lemma 1

$$\textit{There is a path from } s \textit{ to } v \iff \textit{visited}[v] = \textit{true}$$

Lemma 2

$$\textit{All parent edges form a tree}$$

Lemma 3

$$\textit{For each } (u, v) \in E,\ u, v \textit{ are on the same layer or differ by at most 1 layer}$$

Lemma 4

$$A \textit{ node } u \textit{ is in layer } k \iff d(s, u) = k \iff \textit{level}[u] = k$$

Uses of BFS

1. Determine if s, t are connected
2. Determine if the whole graph is connected by determining if there exists a path from one vertex to all other vertices