CS120 Review: Graphs

NJD

2022

Lectures 9 and 10

Definitions —

Directed Graph

- Defined as G = (V, E)
- For each $(u, v) \in E$, $u, v \in V$, u "points to" v

Undirected Graph

- Defined as G = (V, E)
- For each $(u, v) \in E$, $u, v \in V$, u and v "point to each outer"

Digraph

- Simple, unweighted, directed graph

Keywords —

Planar: a graph can be drawn in 2D with no edge crossings.

Walk: a sequence of verticies from s to t

Shortest Walk: the "distance" of s to t (aka, the minimum of the possible lengths)

Theorems and Lemmas —

Shortest Walk Lemma: If w is a shortest walk from s to t, then all of the vertices that occur on w are distinct. That is, every shortest walk is a *path.*

- Suppose we have a shorest walk with repeated vertices. We know there exists a shorter one by simply getting rid of all vertices between the first instance of the repeated vertex and the second for all vertices.

BFS 2-Coloring Theorems BFSColoring on a 2-colorable graph will always return a valid 2-coloring in O(n + m) time

- Think about it: if the graph is 2-colorable, each new frontier will have alternating colors
- Time O(n + m) is achieved through connected components

Algorithms —

Shortest Walk

- Inputs: digraph $G = (V, E)$, verticies $s, t \in V$
- Outputs: shortest walk iff it exists
- Possible solving algorithms:
  1. Exhaustive Search: $(n-1)! \cdot O(n)$
     - Get all walks up to length n-1
     - By Shortest Walk Lemma, our shortest walk must be here
     - Find shortest walk starting from s and ending at t
  2. BFS: $O(n+m)$
     - Initialize array V, where $V[v]$ represents which vertex we got to v from
     - From i in 0 to n-1:
       * If $V[t]$, return the sequence from $V[t]$ down to s
       * Add all new vertices from the current frontier's edges to V if they are not there already
     - Return $\perp$

Coloring

- Inputs: Connceted graph $G = (V, E)$
- Outputs: a "few" coloring of the graph
- Possible solving algorithms:
  1. Greedy: O(n + m)
     - Select ordering of the colors
     - For i in 0 to n - 1
       * Coloring of vertex v_i is the minimum color (i.e., assign each color a number) such that none of its edges also share this color.
     - return this coloring
  2. ***BFS: I think it's $O(n+m)$, but notes aren't too clear...***

- Select ordering of vertices by performing BFS from some vertex v_0
- For i in 0 to n - 1
  - Coloring of vertex v_i is the minimum color (i.e., assign each color a number) such that none of its edges also share this color.
- return this coloring

Connected Components

- Inputs: Undirected graph $G = (V, E)$
- Outputs: a parition of V such that they are connected components
- Possible solving algorithms:

  1. BFS: O(n + m)
     - Select ordering of the colors
     - For i in 0 to n - 1
       - Coloring of vertex v_i is the minimum color (i.e., assign each color a number) such that none of its edges also share this color.
     - return this coloring