CS120 Review: SRE Connected Components

NJD

2022

The Algorithm

Explanation — If we have a graph $G = (V, E)$, we want to partition V such that each pair of vertices that are reachable from one to the other are in the same parition (this is an iff statement). I.e., if we have a connected graph, we return only 1 V: the original V.

The Idea If we do BFS on a random vertex v_0, we can get all connected verticies by implementing improved BFS on just the connected component involving v_0. This is V_0. If there exists any v_i in $V - V_0$, call it again on that. This should be, in total, $\boxed{\text{O(n + m)}}$

The implementation We do this algorithm as a reduction to another algorithm called BFSLabel. First, here is how BFS-CC works:

1. Create an array S, where S[v_i] = $\perp$
2. Create a variable called ℓ, intialize to 0
3. For each v_i:
   (a) if S[v_i] == $\perp$:
      - Oracle call BFSLabel to label each vertex reachable by v_i to the value ℓ

   (b) $\ell += 1$
4. return (ℓ, S)

How does *BFSLabel* work?

1. Label each vertex by some input number ℓ iff it is reachable from starting vertex s; we also input a labeling array S
2. Set $F = [s]$
3. Pretty simple: just do a while loop for if there are vertices reachable from each vertex in F SUCH THAT none of them are labeled l

- If yes: label $S[f_i] = \ell$; update F to be these new verticies
- If no: terminate (we don't have to return since we update S)