CS 161 Fall 2017: Section 5

Farmville

Suppose there is a set of farms that have entered into a cooperative agreement. They built a series of irrigation pipes connecting the farms, with each pipe having a direction of flow (i.e. water can only travel in one, fixed, direction in each pipe). Each farm uses some of the water that passes through the pipes that go through the farm, and each farm also has a well that can contribute water to any irrigation pipes leaving the farm. [Think of this as a way of distributing the risk that certain wells might go dry one year, and others go dry a different year...some years Farmer Joe takes more water from the network than he contributes, some years he supplies more water from his well than he takes.]

(a) After designing the network of (directed) pipes, the farmers want to ensure that water from every farm can get to every other farm. (For example, if there is only one farm whose well is still working, every farmer should still be able to get some of that water via the irrigation pipe network.) The only algorithm that the farmers know is BFS, which runs in time $O(|V| + |E|)$ and takes as input a directed graph $G = (V, E)$ and a source node $s \in V$, and outputs the set of nodes that can be reached from node s. Design a way for the farmers to check whether the desired condition holds for a given proposed pipe network, that uses their BFS algorithm only a *constant* number of times.

(b) Thanks to your solution, the farmers built a successful network, and enjoy many years of steady crops. One year, however, a turnip gets stuck in an irrigation pipe, clogging that pipe; in the week it takes to fix that pipe, Farmer Joe's potatoes all perish from lack of water. The farmer coop forms a new emergency planning committee to design a new, improved irrigation plan. This time, they want a network such that the guarantees of the above part (namely that water from any farm's well can flow to any other farm via the irrigation pipes) will continue to hold even if any single pipe becomes clogged. Describe an algorithm for this part that uses your algorithm from the previous part. Justify the correctness and runtime of the algorithm.

(c) What is the minimum number of irrigation pipes, as a function of the number of farms, n, that are necessary in any network that satisfies the desired property described in part (b)? Prove your answer– both that the claimed number can be achieved, and that no smaller number is possible.

2SAT-SCC

In 2SAT, you have a set of boolean clauses, each containing two variables. Your goal is to set all of these clauses to true, or report that there is no way to accomplish this goal. For example,

$$(x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_3) \wedge (\bar{x}_2 \vee \bar{x}_3)$$

can be satisfied by setting x_1 and x_3 to true and x_2 to false.

(a) We first decompose each clause into its implications: this means that a clause represented by $a \vee b$ would be represented by $\bar{a} \implies b$ and $\bar{b} \implies a$. What are the implications in the clauses above?

(b) We build a graph by creating two nodes for each variable x: x and $\bar{x}$. We then add directed edges based in the implications from each clause. What is the graph representation of the above boolean clauses?

(c) What do SCCs in these 2SAT graphs represent? What if x and $\bar{x}$ were inside of the same SCC?

(d) Assume we have a sink strongly-connected component C from the graph, and that no variable and its negation both appear inside of C. If C has nodes of the form $a, b \ldots z$, what corresponding component do we know must exist in the graph? Is there an edge between these two components?

(e) Given this graph structure and the properties above, develop an algorithm to solve 2SAT.