CS 161 Winter 2021 Section 6

February 18, 2021

Exercise 1

You are given an image as a two-dimensional array of size $m \times n$. Each cell of the array represents a pixel in the image, and contains a number that represents the color of that pixel (for example, using the RGB model).

A segment in the image is a set of pixels that have the same color and are connected: each pixel in the segment can be reached from any other pixel in the segment by a sequence of moves up, down, left, or right. Design an efficient algorithm to find the size of the largest segment in the image.

Exercise 2

In this problem, we design an algorithm to solve the 2-SAT problem. Suppose we have n boolean variables

```
boolean b1
boolean b2
. . .
boolean bn
```

and m conditionals

```
if a1 or a2: // conditional 1
  // do task 1
if a3 or a4: // conditional 2
  // do task 2
. . .
```

where each conditional is the disjunction of two literals `ai` which are each either a boolean variable `(bj)` or some negation of a boolean variable `(not bj)`. For example, one such list of conditionals is

```
if b1 or (not b2):
  // do task 1
if b2 or b3:
  // do task 2
if (not b1) or (not b2):
  // do task 3
```

In this problem, we will devise a polynomial time algorithm to assign all n boolean variables such that every conditional evaluates to true, and every task is executed (if possible). For example, in the above example, the algorithm might output

```
b1 = True
b2 = False
b3 = True
```

Consider a set of $2n$ vertices V where n vertices correspond to the n boolean variables $\{\texttt{b1}, \ldots, \texttt{bn}\}$ and the other n correspond to their negations $\{\texttt{not b1}, \ldots, \texttt{not bn}\}$. Denote the vertex that corresponds to boolean variable `bi` as $v_{\texttt{bi}}$, and the vertex that corresponds to its negation as $v_{\texttt{(not bi)}}$. Construct a directed graph on these vertices such that for every conditional `if c or d`, we add the two directed edges $(v_{\texttt{not c}}, v_{\texttt{d}})$ and $(v_{\texttt{not d}}, v_{\texttt{c}})$. (Note that `c` is any literal and can be a negation of a variable, and `not not c = c`.)

a. Consider an assignment to the boolean variables. Show that all tasks are executed if and only if for every edge $(\texttt{a}, \texttt{b})$ in the graph, if $\texttt{a} = \texttt{True}$, then $\texttt{b} = \texttt{True}$. Hence, we can consider edges of the graphs as "implications": `a` implies `b`.

b. Show that if any boolean variable `bi` lies in the same strongly connected component as its negation `(not bi)` in this graph, then there is no assignment of the n boolean variables that satisfies all m clauses.

c. Show that if there is a path in the graph $(\texttt{a} \to \texttt{b})$ then there is a path $(\texttt{not b} \to \texttt{not a})$.

d. In the following parts, we assume that no literal is in the same strongly connected component as its negation. Consider one strongly connected component $S \subset V$ of this graph. Show that the negations of every literal $a \in S$ are contained in a single other strongly connected component $\bar{S} \subset V$.

e. If a particular strongly connected component S is a sink node of the SCC meta-graph (i.e., there are no edges from any vertex in S to any vertex outside S), what do we know about $\bar{S}$?

f. Design an algorithm that finds an assignment that leads to an execution of all tasks (assuming no literal is in the same strongly connected component as its negation). Prove that your algorithm gives a valid truth assignment to each literal that satisfies every conditional, and show that it has running time polynomial in n and m.