TCSS 343 - Week 8

Jake McKenzie

September 4, 2018

Graph Algorithms

"If youve never missed a flight, youre spending too much time in airports."

...

Umesh Vazirani

"Programming has things called "threads" and things called "strings" and they somehow have fuck all to do with each other."

...

Ramsey Nasser

"We grow in direct proportion to the amount of chaos we can sustain and dissipate."

...

Ilya Prigogine

So when I was taking TCSS 343 I identified two core ideas of Dijkstra's algorithm. I'm going to present them to you for you to think about, then ask you to complete the algorithm.

How does Dijkstra's algorithm find new paths and do the **relaxation step***?*

In which order does Dijkstra's algorithm **process** *the vertices one by one?*

0. Fill in the rest of the algorithm for relaxing an individual edge.

```
Relax(u,v) {
    if ( ________ < d[v]) {
        ________ = d[u] + ________;
        pred[v] = ________;
    }
}
```

Remark: The predecessor pointer pred[] is determing the shortest path.

1. Part of the answer to the second question is that we store the vertices of V/S(set division), where S $\subseteq$ V for which we know the true distance, in a priority queue (typically), where the key value of each vertex v is d[v]. That's all well and good but what are the following runtimes of the operations Insert(), Extract_Min() and Decrease_Key()?

These operations run in ________ time.

2. The initialization step of Dijkstra's algorithm runs in ________ time? (**Hint:** for each u $\in$ V)

3. Show that $VO(1+\log V)+O(E)+O(E\log V) \in O((V+E)\log V)$.(Dijkstra's algorithm runtime w/ a priority queue...**Bonus question:** Can you figure out the overall runtime of a sparse vs dense graph? All runtimes for bonus question can be expressed in terms of V.)

4. Consider Kruska's algorithm for the graph $G = (V,E)$ with edges $\{A,B\}$, $\{B,C\}$, $\{A,D\}$, $\{B,D\}$, $\{C,E\}$, $\{B,E\}$, $\{D,E\}$, $\{D,F\}$, $\{F,E\}$, $\{F,G\}$, $\{E,G\}$. If the edge weights are:

$w(\{A,B\}) = 7$, $w(\{B,C\}) = 8$, $w(\{A,D\}) = 5$, $w(\{B,D\}) = 9$,
$w(\{C,E\}) = 5$, $w(\{B,E\}) = 7$, $w(\{D,E\}) = 15$, $w(\{D,F\}) = 6$,
$w(\{F,E\}) = 8$, $w(\{F,G\}) = 11$, $w(\{E,G\}) = 9$

5. Now use Prim's algorithm on the same graph G. Did you obtain the same results?

For the next few problems state whether these desiderata are true or false and justify your work.

6. If the priority queue in Dijkstras algorithm is implemented using a sorted linked list (where the list is kept sorted after each priority queue operation), then Dijkstras algorithm would run in $O((E + V)\log V)$ time.

7. Given a graph with unique positive edge weights, the shortest path will be contained within the minimum spanning tree of the graph.

8. If $P \neq NP$ then every problem in NP requires exponential time.

9. If problem ψ is NP-hard and $\sigma \leqslant_p \psi$ then σ is NP-hard.

A. If problem σ is NP-hard and $\sigma \leqslant_p \psi$ then ψ is NP-hard.

B. If a problem σ is in P then $\sigma \leqslant_p \psi$ for every problem ψ in NP.

C. If G is a weighted graph with n vertices and m edges that does contain negative-weight cycle, then for every vertex $v \in G$ the shortest path from $v \to t \in G$ containing n edges is strictly shorter than the shortest path from $v \to t \in G$ containing $n - 1$ edges.

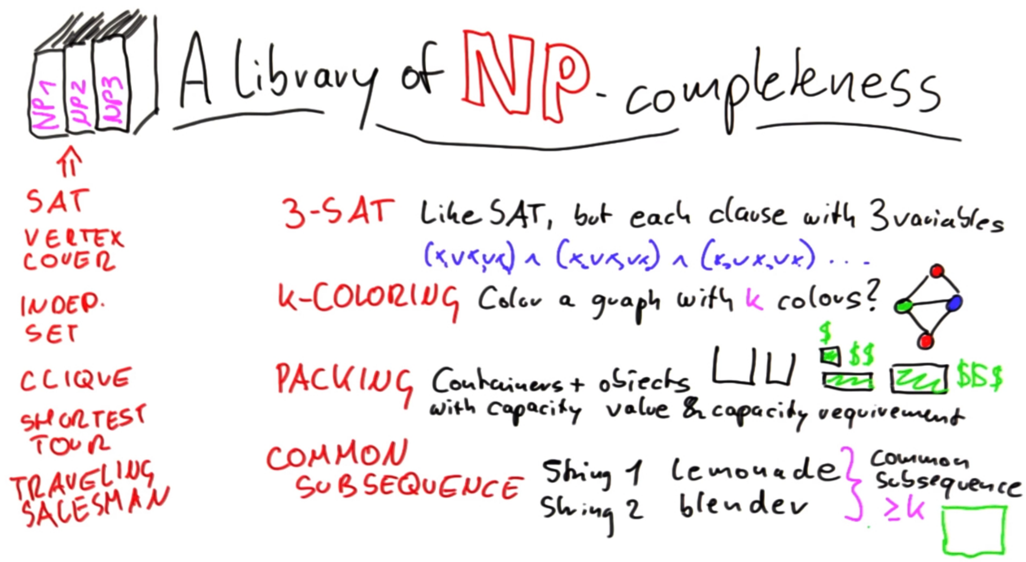
There are n cities in a country.

Every two cities u and v are connected by a bi-directional highway that takes $l(u, v)$ hours to drive through.

A car with gas tank capacity c can only travel for c hours on a highway and has to be refueled at the cities.

D. Give an $O(V^3 \log V)$ modification to Dijkstras algorithm to compute, simultaneously for all pairs (u, v) of cities, the minimum gas tank capacity needed to go from $u \rightarrow v$.

E. There are many problems that are NP-Complete with similar problems are P. For example vertex cover is NP-Complete while edge cover is P, Hamiltonian path is NP-Complete while Euler path is P, 3-Sat is NP-Complete while 2-SAT is P, etc. Make a conjecture for a problems covered in class or seminar that is in P with a corresponding problem that is NP-Complete. If you can find another P time algorithm that is similar to the NP-Complete algorithms stated here that's fine. Below is a list of some more NP-Complete problems. Why did you make this conjecture?



Please spend as much time on this problem as the previous. It's important to make the connection of these concepts for yourself.

F. Given an undirected graph G, design an algorithm to list the vertices in each connected component of G seperately.

10. Give an $O(V)$ time algorithm to determine whether a connected undirected graph contains a cycle.

The vertex cover problem states given $G = (V, E)$, we want to know the minimum size of the vertex cover $C \subseteq V$, where C is a vertex cover if every edge $e = (u, v) \in E$ has at least of of it's endpoints u or $v \in C$. (The problem generally is NP-Complete but given certain simple graphs the problem is not nearly as hard to solve).

11. Suppose that G is a tree. Give a polynomial time algorithm to find the minimum vertex cover. (Hint: Consider being greedy and remember that you can choose the optimal solution you like.)