22 (100 PTS.) COVID-19

COVID-19 is the most major pandemic of our lifetime! As of 03/23/2020 3 pm CST, more than 370,000 individuals have been affected and the numbers are increasing exponentially fast everyday. Please stay at home and avoid social contact.

Graphs are an extremely powerful tool in analyzing and tracking the spread of viruses. Suppose there are n people in a given community, $P_1, P_2, \cdots, P_n$. You are given the time at which any pair of individuals came into contact during some observation period. Hence, the data is a sequences of ordered triples (P_i, P_j, t_k) which means that person P_i was less than 6 feet away from person P_j at time t_k. Furthermore, if a person P_i carrying the corona virus comes in contact with another person P_j not carrying the corona virus at time t_k, then the virus is transfered to person P_j at time t_k.

The infection can spread from one person to another across a sequence of contacts, provided that no step in this sequence involves a move backward in time. Hence, if P_i contracts the virus at time t_k, and the data you are given contains triples (P_i, P_j, t_k) and (P_j, P_q, t_r), where $t_k \leq t_r$, then P_q will contract the virus via P_j. (Note that it is okay for t_k to be equal to t_r; this would mean that P_j was less than 6 feet away from both P_i and P_q at the same time, and so a virus could move from P_i to P_j to Pq.)

For example, suppose $n = 4$ and person P_1 contracts the virus at time $t = 2$.

If the data contains the triples:

$$(P_1, P_2, 4), (P_2, P_4, 8), (P_3, P_4, 8), (P_1, P_4, 12)$$

Then, P_3 would contract the virus at time 8. However, if the data contains the triples:

$$(P_2, P_3, 4), (P_1, P_4, 8), (P_1, P_2, 12)$$

Then, P_3 would not contract the virus during the observation period.

For simplicity, you can assume that the triples are given to you in a sorted order of time. You can also assume that each pair of individuals come into contact at most once during the observation period and sick individuals remain sick for the entire observation period.

Design an algorithm that given a sequence of m triples and the observation that person P_x contracted the virus at time t_x, finds all the people who would contract the virus during the observations interval and the time at which they contracted the virus. (**Hint:** Build a graph. What are the vertices? What are the edges? What problem is this?)

Solution:

We construct a directed graph G as follows:

- For every triple (P_i, P_j, t_k), we create vertices (P_i, t_k) and (P_j, t_k).
- We add directed edges between these two nodes in both directions.
- If this is not the first triple involving P_i, we add a directed edge from (P_i, t) to (P_i, t_k) where t is the timestamp in the preceding triple that included P_i. In other words, for all vertices (P_i, t) already in the graph, add an edge from the one with the largest t that is smaller than t_k. Similarly, for P_j.

The graph has $O(m)$ vertices and $O(m)$ edges. Since we are given the sequence of triples in sorted order, we can construct the graph in $O(m + n)$ time. To do so, we maintain an array of size n pointing to linked lists associated with each person P_x. Whenever a vertex (P_i, t_k) is created, we append it to the list associated with P_i. Since, we are given the triples in sorted order with respect to t, we can directly find (P_i, t) as the last appended vertex to the list and add an edge from it to (P_i, t_k).

Given that person P_x contracted the virus at time t_x, we walk through the list for P_x until we get to the last node (P_x, t') such that $t' \geq t_x$. We run basic graph search (BFS or DFS) from (P_x, t') and find all the nodes that are reachable from it. We then scan these nodes and for each person P_y such that (P_y, t_k) is reachable, we maintain t_{min}: the minimum t_k.

Correctness: First, we claim that if there is a path from (P_x, t') to (P_y, t_{min}) then P_y would contract the virus by time t_{min}. To see this, we simply have the virus moves between P_i and P_j at time t_k, whenever an edge from (P_i, t_k) to (P_j, t_k) is traversed by graph search. This is a feasible sequence of virus transmissions that results the virus leaving P_x at time t_x or $t' \geq t_x$ and infecting P_y at time t_{min}.

Conversely, suppose there were a sequence of virus transmissions that results in the virus first leaving P_x at time t_x or later and arriving at P_y at time t_{min}, Then, we can build a path in our graph as follows. We start at node (P_x, t') and follow edges to when the virus first leaves P_x. For each time that the virus moves from P_i to P_j, there must be a edge from (P_i, t_k) to (P_j, t_k) which we can add to the path. If the virus moves out of P_j at time $t \geq t_k$, we add the sequence of edges (P_j, t_k) to (P_j, t). When the virus, first arrives at P_y at time t_{min}, we will have added the node (P_y, t_{min}) to the path. Hence, there is a sequence of edges forming a path from (P_x, t') to (P_y, t_{min}).

Runtime: Running graph search on G and finding the minimum takes $O(m)$ and constructing the graph takes $O(m + n)$. Hence, the algorithm takes $O(m + n)$.

Rubric: 100 points: standard graph-reduction rubric.

- -5 for $O(m)$ instead of $O(m + n)$ algorithm.
- -10 for assuming (P_x, t_x) is a node in the graph.
- -40 for $O(m^2)$ algorithm that assigns every triple to a vertex.

23 (100 PTS.) Racetrack

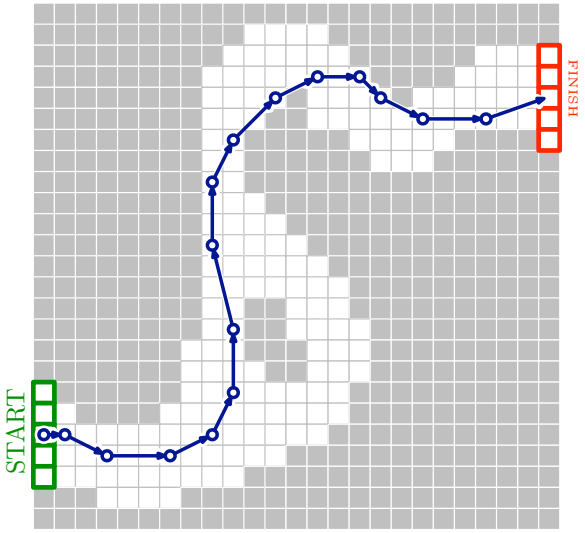
Consider the following two-player paper-and-pencil racing game. The game is played with a track drawn on a sheet of graph paper. The players alternately choose a sequence of grid points that represent the motion of a car around the track, subject to certain constraints explained below.

Each car has a *position* and a *velocity*, both with integer x- and y-coordinates. A subset of grid squares is marked as the *starting area*, and another subset is marked as the *finishing area*. The initial position of each car is chosen by the player somewhere in the starting area; the initial velocity of each car is always $(0, 0)$. At each step, the player optionally increments or decrements either or both coordinates of the car's velocity; in other words, each component of the velocity can change by at most 1 in a single step. The car's new position is then determined by adding the new velocity to the car's previous position. The new position must be inside the track; otherwise, the car crashes and that player loses the race. However, it is not necessary for the line between the old position and the new position to lie entirely within the track. The race ends when the first car reaches a position inside the finishing area.

Suppose the racetrack is represented by an $n \times n$ array of bits, where each 0 bit represents a grid point inside the track, each 1 bit represents a grid point outside the track, the "starting area" is the first column, and the "finishing area" is the last column.

Describe and analyze an algorithm to find the minimum number of steps required to move a car from the starting line to the finish line of a given racetrack. (**Hint:** Build a graph. What are the vertices? What are the edges? What problem is this?)

velocity	position
$(0, 0)$	$(1, 5)$
$(1, 0)$	$(2, 5)$
$(2, -1)$	$(4, 4)$
$(3, 0)$	$(7, 4)$
$(2, 1)$	$(9, 5)$
$(1, 2)$	$(10, 7)$
$(0, 3)$	$(10, 10)$
$(-1, 4)$	$(9, 14)$
$(0, 3)$	$(9, 17)$
$(1, 2)$	$(10, 19)$
$(2, 2)$	$(12, 21)$
$(2, 1)$	$(14, 22)$
$(2, 0)$	$(16, 22)$
$(1, -1)$	$(17, 21)$
$(2, -1)$	$(19, 20)$
$(3, 0)$	$(22, 20)$
$(3, 1)$	$(25, 21)$



A 16-step Racetrack run, on a 25×25 track. This is *not* the shortest run on this track.

Solution:

Let $T[1..n, 1..n]$ be the input bitmap. We construct a directed graph G as follows:

- G has a vertex for each integer vector $(x, y, \Delta x, \Delta y)$ that represents a legal position and velocity for the car. The integer vector (x, y) is a legal position if and only if $T[x, y] = 0$ (and therefore $1 \le x \le n$ and $1 \le y \le n$). By the following claim, $(\Delta x, \Delta y)$ is a legal velocity if and only if $-\sqrt{2n} \le \Delta x \le \sqrt{2n}$ and $-\sqrt{2n} \le \Delta y \le \sqrt{2n}$.

  Claim 8.1. *The horizontal/vertical speed of the car cannot exceed $\sqrt{2n}$.*

  Proof: Suppose the car starts in column 0 of the grid, with velocity zero, and accelerates as quickly as possible to the right. After t steps, the car has velocity $(t, 0)$ and lies in column $\sum_{i=1}^{t} i = t(t+1)/2$. Because the car cannot go past the right edge of the grid, we must have $t(t+1)/2 \le n$, which implies that $t \le \sqrt{2n}$. Similar arguments imply that the car's vertical speed cannot exceed $\sqrt{2n}$. ■

- G has a directed edge for each legal move by the car. Specifically, G contains the directed edge $(x, y, \Delta x, \Delta y) \to (x', y', \Delta x', \Delta y')$ if and only if both endpoints are legal vertices and all of the following conditions are satisfied:

$$\begin{aligned} x' &= x + \Delta x & \Delta x' &\in \{\Delta x - 1, \Delta x, \Delta x + 1\} \\ y' &= y + \Delta y & \Delta y' &\in \{\Delta y - 1, \Delta y, \Delta y + 1\} \end{aligned}$$

- G also has an artificial starting vertex s, with $O(n)$ outgoing edges to every vertex of the form $(1, y, 0, 0)$ such that the point $(1, y)$ is in the starting area.
- Finally, G also has an artificial target vertex t, with $O(n^2)$ incoming edges from every vertex of the form $(n, y, \Delta x, \Delta y)$ such that the point (n, y) is in the finishing area.

The graph G has at most $n \times n \times 2\sqrt{2n} \times 2\sqrt{2n} + 2 = 8n^3 + 2$ vertices. Hence, $|V| = O(n^3)$. From each vertex other than s, there are at most 9 outgoing edges. Hence, $|E| = O(n^3)$.

By construction, every directed path from s to t in G represents a sequence of steps that begins with the car at the starting line with velocity $(0, 0)$ and ends with the car at the finish line. Moreover, a path of length ℓ corresponds to a sequence of $\ell - 2$ steps. Thus, the shortest sequence of steps corresponds to the ***shortest path*** from s to t in G. We compute this shortest path in $O(V + E) = \boldsymbol{O(n^3)}$ ***time*** via ***breadth-first search***.

Rubric: 100 points: standard graph-reduction rubric.

- -10 for using Dijkstra's algorithm instead of BFS.
- -20 for only showing $O(n^4)$ time instead of $O(n^3)$ time for this algorithm for whatever reason:
  - Not accounting for $|\Delta x| \le \sqrt{2n}$ and $|\Delta y| \le \sqrt{2n}$
  - Not adding a start vertex and running BFS for each start position.
  - Some other inefficient but correct algorithm
- -10 for each factor of n in the running time above n^4.

24 (100 PTS.) **374-Walk**

Consider a directed graph G, where each edge is labeled with a symbol: either **3**, **7** or **4**.

A walk in G is called a ***374 walk*** if its sequence of edge symbols is **3, 7, 4, 3, 7, 4, 3, 7,** $\cdots$. Formally, a walk $v_0 \to v_1 \to \cdots \to v_k$ is a ***374 walk*** if, for every integer i, the edge $v_i \to v_{i+1}$ is labeled with symbol **3** if $i \bmod 3 = 0$, **7** if $i \bmod 3 = 1$, and **4** if $i \bmod 3 = 2$.

Describe an efficient algorithm to find all vertices in a given labeled directed graph G that can be reached from a given vertex v through a ***374 walk***.

Solution:

Given the edge-labeled graph $G = (V, E)$, we construct a new directed graph G' with vertices $V \times \{\mathbf{3}, \mathbf{7}, \mathbf{4}\}$ and the following edges:

- $(v, \mathbf{4}) \to (u, \mathbf{3})$ for every edge $v \to u \in E$ that is labeled **3** ;
- $(v, \mathbf{3}) \to (u, \mathbf{7})$ for every edge $v \to u \in E$ that is labeled **7** ;
- $(v, \mathbf{7}) \to (u, \mathbf{4})$ for every edge $v \to u \in E$ that is labeled **4** ;

Every walk in G' that starts at a vertex $(v, \mathbf{4})$ corresponds to a 374 walk in G that starts at the corresponding vertex v, and vice versa. Thus, our task is to determine which vertices are ***reachable*** in G' from vertex $(v, \mathbf{4})$. We can solve this problem by ***breath/depth/whatever-first search*** in $O(V' + E') = \boldsymbol{O(V + E)}$ ***time.***

Rubric: Standard graph reduction rubric