note 187 views

Soooo You Messed Up on Exam 2...

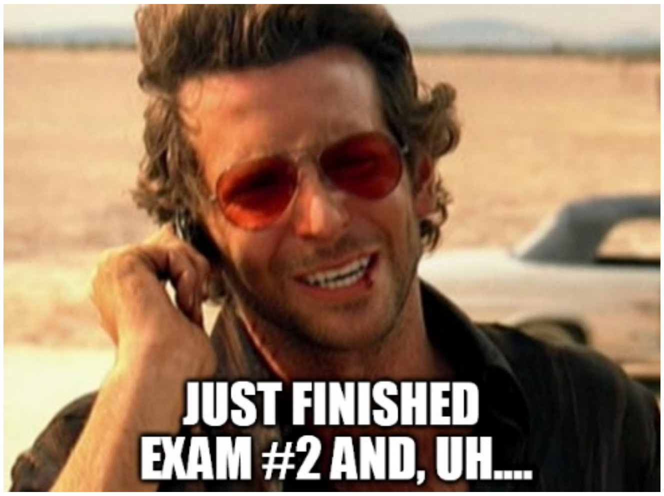
Hi All,

Soooo you messed up on exam 2. What cringe-worthy things did you do wrong that you realized as you were scanning / 30 mins after the test?

For me:

1. I realized that I said "if edge e` is less than e* remove" instead of "if edge e` is **greater** than e* remove"...one word screw-up really destroyed that. Not that I think I had a great solution for #2 anyway....
2. #3.b I screwed up returning the true/false. I think I did it correctly otherwise but I returned true when I should have returned false and vice versa....

What stupid mistakes did you make on exam 2 that haunted you in your dreams last night?



exam

Updated 13 hours ago by Brent Wagenseller

followup discussions *for lingering questions and comments*

● Resolved ○ Unresolved

Nolan Capehart 12 hours ago I forgot to write my name.

Brent Wagenseller 12 hours ago doh!

Nolan Capehart 12 hours ago Yup. When I uploaded the scan, I saw the problem, but I remembered that they used this as an example of how we should not write anything on the paper after showing it to the camera.

Ronny A. Peña 12 hours ago Realizing you misread the problem statement after you almost completed a solution. lol. Luckily, I had enough time to erase and redo.

Brent Wagenseller 12 hours ago oh no ronny. I print out the exam portion twice and just use the problem space for my scrap paper (instead of printing the designated scrap pages); in the event I would have to erase a whole page I just replace with the back-up scrap paper I made.

Scott Jensen 12 hours ago I made a greater than/less than mistake on that problem as well. Luckily I caught it when I read through my answers before submitting (still had time left on the clock)

Ronny A. Peña 12 hours ago Good idea Brent. I will do the same for next time. I didn't had to erase the whole thing, just a few lines.

Brian Xia 11 hours ago For p4, I checked dist(v, z*) > dist(z*,v), missing a factor of two. Realized it when I was scanning it, which left a very sour aftertaste :((

Carl Shek 10 hours ago Wrote the correct runtime for Dijkstra's algorithm in scrap paper, but in the actual problem, I wrote O(|V| * |E| log |V|) instead of O((|V| + |E|) log |V|)

Isaac Gutierrez 6 hours ago Wrote $O(m \log n)$ for running time of MST instead of $O(m \log m)$, forgot to add my name also.

● Resolved ○ Unresolved

John-Peter 11 hours ago

I think for question 2 I didn't actually use an existing algorithm as a blackbox to explain a particular step and instead just described what to do intuitively, which in hindsight seems like I might lose a lot of points for. Really dumb mistake, I hope I don't get burned too badly.

Nolan Capehart 8 hours ago As far as I know right now, there really wasn't a way to do it with one of the other algorithms as a completely-black box. Maybe I overlooked something, though.

Patrick Miller 8 hours ago It wasn't entirely clear to me either, but I solved it by creating a cycle. In order to find the edges in the cycle, I used DFS as my black box. Everything else was either O(1) or simple loops.

Gautam Venkataramanan 8 hours ago +1 I did same solution as Patrick.

Scott Jensen 8 hours ago +1 I found a path from x to y using DFS creating a cycle with every edge on that path and the new edge.

Eric Eidson 7 hours ago I used DFS to find the path from y to z in graph T, which would be coincident with the cycle. Since T is a tree, there is only one path from y to z. In the process you also find the heaviest edge in the path. Add the new edge and delete the heavy edge and you have T'.

Nolan Capehart 7 hours ago I also used DFS to find the cycle. However, the problem is what to do after you determine there's a cycle. (Of course, we already know there is one, but what I mean is that in DFS you detect the cycle.) How exactly can you then get the heaviest edge on the cycle? Basically you have to have a stack of some sort, which DFS does have, but then as you're popping the stack, you have to pay attention to which edge is the heaviest. That kind of sounds to me like modifying the algorithm, but I strongly suspect that's the expected answer.

In the last couple of minutes, I also scribbled that you could do this with just Explore() in the same way, before adding the new edge. If you start Explore at y, when you see y again, you've found the cycle. There is no need to keep pre and post-order numbers, in other words.

Nolan Capehart 7 hours ago I'm probably just over-thinking things, though.

Anshul Mohnot 7 hours ago If you are adding an edge e* from y to z. Just use DFS to find the path from y to z in the MST T. On this path find the edge with the largest weight, if this weight is greater than b, remove this edge and replace it with e* and you get your new MST T'
The runtime of this algorithm is O(n + m), for a tree m is n-1, run time is O(n)

Patrick Miller 7 hours ago Finding the heaviest edge from the list of edges is just a simple loop, which can be considered as the function that you use to modify the output of the blackbox to fit the problem. No modification of DFS/Explore is needed, just return the edges visited.

Nolan Capehart 7 hours ago I missed the fact that you can reduce it to O(n). That's a good point.

Scott Jensen 7 hours ago Yeah, you don't need to modify an algorithm but you do need to have your own "code" to use it in a smart way. I realized for some reason I just replaced an edge heavier than the new edge, not the heaviest. I even thought about doing heaviest, but for some reason I was getting so caught up in thinking through the cycle logic I forgot to make it a MST. Mine is a lighter tree, not the lightest. Hopefully that isn't too costly.

Eric Eidson 7 hours ago I vaguely recall that e* was specified to be lighter than any edge in T, so it was not necessary to ensure it was lighter than the edge it replaced. Hmm.

Scott Jensen 7 hours ago I don't think it said that. In fact I'm pretty sure I checked it. We just know e* is lighter than it used to be. Was a now is b < a

Nolan Capehart 7 hours ago I remember it the same way Scott does.

Gautam Venkataramanan 6 hours ago I added e* to create cycle. Then walked the cycle to find an e such that a >= w(e) >= b. Then replaced e with e*.

Wes Heath 5 hours ago I didn't use any of the algorithms as a black box either. I thought about DFS, but didn't use it because it was unclear to me how DFS could identify a cycle without modification. DFS from y would find all of the vertices on the path to z, but it would also identify vertices that weren't on that path and therefore not part of the cycle that adding the new edge created. I just scanned the list of edges that were given to us in the problem setup.

Santiago L. Valdarrama 5 hours ago

- I used BFS to find a path from y to z.
- Then computed the heaviest edge e' in that path.
- Then if w(e') >= w(e*), T' = T u e* - e'
- If w(e') < w(e*), T' = T

Arvind Kumar Sharma 2 hours ago I was skeptical about using DFS to return cycle but that looked only solution less costly in terms of time than anything else so went ahead with that anyways... Feeling relaxed to see that so many people followed the same :-)
My steps:

1. Run DFS on TUe* to find the cycle C - O(n) time
2. Find heaviest edge in C - O(n) time
3. Remove heaviest edge in C - O(1)

So solution is O(n) time

● Resolved ○ Unresolved

David Duffy 9 hours ago

I might not have written things in the desired wording format, that seems to be one of my undoings in this class. But I sure hope I didn't screw up like exam 1 with dumb mistakes.

● Resolved ○ Unresolved

Wes Heath 5 hours ago

I think one of the main issues I had with Exam 2 is that it was much less clear what I needed to be focusing on than it was with Exam 1. Like the fact that DPV Chapters 4 and 5 were basically required reading, despite not being on the schedule. Or the fact that we apparently only needed to read a portion of Chapter 7, though we were never explicitly told which portions, so I totally skipped the bipartite graph section which basically was the answer to question 1. I felt very prepared going into Exam 1 and knew what to expect - that wasn't really the case with Exam 2.