Name: Michael Rogers

ID: 105667404

CSCI 3104, Algorithms **Profs. Grochow & Layer**

Explain-It-Back 8 **Spring 2019, CU-Boulder**

You are collaborating with a geology team that is using a rover to explore lava fields. They have mapped out the surface of the lava field as a grid where each edge is annotated with the likelihood that the robot can successfully navigate the corresponding terrain. This likelihood integrates physical properties such as surface temperature and relief. Their current algorithm finds a path by considering all of the edges at its current position, then all of the edges that are one step away, two steps away, and so on until they reach the desired destination. Unfortunately, this process takes so much time to complete that the physical properties of the edges change before a route can be calculated rendering the path useless. You have the insight that the robot does not need the best path, just one that can be calculated quickly and has a reasonable likelihood of success. Help your team understand the issues with the current solution, and how a simple algorithm change could help.

Name: Michael Rogers
ID: 105667404

CSCI 3104, Algorithms **Profs. Grochow & Layer**
Explain-It-Back 8 **Spring 2019, CU-Boulder**

Hello geology colleagues,

It sounds like you guys are working on an extremely interesting and important project. I am very happy that you have trusted me to help you solve this problem. I think I can provide some insight that should help us resolve the delay that you are seeing in processing. I'd first like to discuss the issues with your current algorithm, and then we can look at my recommendation for a solution. Let's begin with the problems in your current algorithm. Right now you are using something we in the computer science community call a breadth first search for a path. This is the process in which you search through the first tier paths, the second tier paths, and so on until you reach the terminal point. While this is an great way to search through paths, it can be very taxing on computing power. This delay occurs because you are searching through every possible path to the terminal point. Let's move on to my recommendation to make the your existing algorithm more efficient. I suggest you simplify your algorithm just a little bit. Instead of using the breadth first search, I suggest you use a simpler algorithm that doesn't require so much computing power. Instead of looking at every possible path I suggest that we use something called a depth first search. This is the process by which we follow a specific path until we reach the terminal point. By using this method we don't have to search every possible path. If we follow a certain path until we find the terminal point, then we can significantly reduce the amount of time it takes to find the most efficient path, or a path that has the most likelihood of finding the terminal point. If you have any questions about how to implement this algorithm, feel free to ask me. Good luck!

Sincerely,
A Computer Scientist you know