Networks and Complexity

Solution 2-11

Ex 2.11: Time travel [4]

Suppose there is a magical portal in Bree. Whoever steps into the portal magically transported to Isengard. The magical travel isn't instantaneous, it's actually even faster than that. People emerge in Isengard two days before they step into the portal in Bree. Does this present a problem for Dijkstra's algorithm? Can we still use it?

Solution

Dijkstra's algorithm can be applied to a wide variety of problems which go beyond the original context. However, when time travel comes into play we have to be very careful. To understand what can go wrong it is useful to distinguish between two scenarios.

First, consider that we could try to take the portal to Isengard, then walk back to Bree and take the portal again. In this example the distance is big enough that we would spent more time walking back then we gain from taking the portal. If the numbers were different or we had a faster mode of travel (good idea actually...) we could do multiple trips through the portal to travel back in time as far as we want. This makes the whole shortest path question meaningless as we can arrive at every destination as early as we want.

Let's now focus on the second scenario where we cannot exploit the portal to go arbitrarily far back in time. In this case we are almost back to a normal shortest path problem, but there is one important twist. Remember that after updating from Hobbiton we argued that we can now be sure that we have found the shortest path to Bree, because every other path would need to go via some other city (Isengard) that is father from Hobbiton then Bree and hence cannot be a shortcut. Now that time travel is a possibility we can't be sure about that anymore. This means that there might be some cases where we need to update from the same city multiple times. In this case we need to keep an explicit list of places from which there could still be updates possible. Whenever we find a new shortest route to a place it gets added (or re-added) to the list. The places from which we have updated get removed, until the list is eventually empty. If there isn't a loop in which we can arbitrarily far back in time the algorithm will eventually converge, and we can be sure that we have found all shortest paths.

As a final comment also notice that the time travel presents us with a completely different philosophical problem: What were we actually asking for? I mean, when we are looking for the shortest path do we want to arrive at the destination at the earliest possible date or do we want the path where we spent the least time travelling? When time travel is a possibility those two options aren't the same anymore. The magical portal might take us back two days in the calendar, but it would not undo two days worth of sore feet from walking. So if we settle for the second alternative, and ask for the path that requires us to walk the shortest time, the two days that the portal subtracts from our calendars aren't even relevant, and we can apply plain old Dijkstra to find the solution.