CENG 315

Algorithms

Take Home Exam 3
Complexity Analysis

Student Information

Full Name : Beyza Bütün
Id Number : 2171411

In this homework, I tried to find the largest ammo when the DOOM guy saved the Bunny. To be able to do that, I used the Dijkstra Algorithm. Because we should reach three points which are the key room, the scientist' room and the room of the chamber, I did this in three step.

In my first step, I started from the start point which is 1. Then, my final point was the room of the key. In this path, I didn't pass through the room of the scientist and chamber, because I didn't have a key. While doing this, I aimed to find the least ammo amount which the guy could lost.

In my second step, I started from the room of the key. Then, my final point was the room of the scientist. In this path, I didn't pass through the room of the chamber.

In my final step, I started from the room of the scientist, and my final point was the room of the chamber.

When doing them, I considered the rooms which is unlock in even or odd terms.

Actually, my code is successful for the case when the ammo is not in the adjacency rooms. Otherwise, it is not successfil.

To be able to show the complexity of the code, first I will write my pseudocode. (it contains main points)

notes: per is the period which starts from 1. init is the ammo amount losted and starts from 0.

```
1       for j=0 to j=2
2               for i ∈ nRooms (v ∈ V)
```

```
3                   do periods[i] ← per
4                   do dist[i] ← ∞
5                   do visited[i] ← false
6             do parent[allKeys[j]-1] ← -1
7             do dist[allKeys[j]-1] ← init
8             for k=0 to nRooms (v ∈ V)
9                   u ← findMin(dist)
10                  per ← periods[u]
11                  visited[u] ← true
12                  if (u+1)==allKeys[j+1]
13                        do init ← dist[u]
14                        break
15                  for v=0 to nRooms (v ∈ V)
16                        amountAmmo ← ammoAmount(roomYammo,v+1)
17                        isForbidden ← is_Forbidden(per+1,evenPer,oddPer,v+1)
18                        do if isForbidden
19                              then continue
20                        do if !visited[v] && adj[u][v] && dist[u]!= ∞ &&
(dist[u]+adj[u][v]-amountAmmo) < dist[v]
21                              parent[v] ← u
22                              periods[v] ← per+1
23                              dist[v] ← dist[u] + adj[u][v] - amountAmmo
```

Assume that, the graph is G(V,E), and V is the number of rooms which is nRooms in my code, and E is the number of edges.
I will do the complexity analysis by using V instead of nRooms.

The inner most loop (bw 15 and 23) is looping by V times. Because, it passes through all the rooms to find the adjacency of u.
The second for loop (bw 8 and 23) is looping by V times. Because again it passes through all the rooms to visit all rooms by once.
The first loop (bw 1 and 23) is looping by 3 times. Because there is 3 starting points and 3 ending points.

So finally it is looping $3V^2$ times. Therefore the complexity is $\Theta(V^2)$ like the Dijkstra algorithm. It means this is polynomial.