13주차 예비 보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20221559 이름: 박준우

1. dfs와 bfs 모두 visited 배열을 이용하여 한 번 방문한 노드는 체크해 준 후 다시 방문하지 않으므로 미로가 N행 M열이라고 했을 때 최대 NM개의 노드를 방문한다. 각 노드 내에서의 코드는 상수시간에 실행되므로 O(NM)이 된다.

2. iterative dfs, bfs를 이용하여 경로를 찾는 법

격자 그래프 형식으로 상하좌우 이동 가능한 지에 대한 여부를 bitMask로 가지고 있는 자료구조를 이용해 미로를 표현했다.

2-1. Iterative dfs using stack

처음에 스택 두개가 필요하다. 탐색할 때, 지나온 경로들의 좌표를 담을 수 있도록 pair<int, int>를 담을 수 있는 스택을 두 개 선언하여서, while loop를 돌리는데, stack의 꼭대기에 있는 위치가 도착지점일 때 까지 실시하면 되고, 현재 위치에서 네 방향을 차례로 탐색하는데, 이곳이 미로 안이고, 벽이 뚫려 있어 갈 수 있으며, 방문하지 않은 곳일 경우에 방문처리를 해 주고, 스택에 쌓아준 후, 다음 while loop에서 해당 위치를 현재 위치로 고려하게 하면 된다. 만약 현재 위치에서 갈 수 있는 곳이 없다면 stack에서 현재 위치를 빼 주면 이전 위치로 돌아가서 다른 방향을 탐색하게 된다.

만약 현재 위치가 도착지점이면, 스택에 들어있는 위치들이 경로의 역순이므로, 결과 스택에 차례로 쌓아주면 된다.

2-2. bfs using queue

bfs의 경우에는 목적지에 도달할 수 있는지에 대해서 확인하는 것은 간단하나, 그냥 진행한 후에 이 경로를 추적하기 위해서 어떤 방법이 필요하다.

강의 자료에 나온 대로, spanning tree의 관점에서 각 위치의 tree에서의 depth를 bfs 중에 배열에 기록해 놓으면, 목적지에서부터 하나씩 작은 노드로 타고 가면 경로를 역추적할 수 있을 것이다.

Bfs 구현 방법은 Queue에 시작 지점을 넣고, 각 지점에서 방문하지 않고, 벽이 뚫려 있어 이동할 수 있는 칸에 대해서 전부 queue에 넣고, 이 지점에 현재 지점의 depth + 1의 값을 넣어 줌으로 spanning tree의 높이를 모두 기록 할 수 있을 것이다.