13주차 예비보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211522 이름: 김정환

**1.**

.................

우선 DFS는 가장 최악의 경우에 모든 노드를 방문하고 각 노드의 모든 경로를 지나게 된다. 노드의 총 개수는 주어진 미로의 row, col을 N, M이라고 할 때 총 N\*M개 존재한다. 경로는 미로에서 움직일 수 있는 범위는 상하좌우만 존재하므로 각 노드당 최대 4개 존재하는데, 여기서 cycle을 만들지 않도록 하는 경로의 최대 개수는 N\*M - 1이다. 따라서 모두 지나갈 때의 경우의 수는 2 \* N \* M - 1이므로 시간 복잡도는 O(N\*M)이 된다.

BFS 역시 가장 최악의 경우가 모든 노드를 방문하고 각 노드의 모든 경로를 방문하는 경우이다. 따라서 구하는 과정은 위와 똑같으므로 BFS의 시간 복잡도 역시 O(N\*M)이 된다.

................

**2.**

.......................

1. DFS

DFS는 강의자료에 나온 대로 stack을 사용하여 이전 노드로 쉽게 돌아갈 수 있도록 하여 구현할 수 있다. 필요한 변수로는 노드별 방문 여부를 저장할 array가 필요하고, 현재 노드를 저장하는 변수와 right와 down 방향의 노드를 저장하는 변수가 필요하다. 이를 통해서 maze의 row = 1, col = 1인 지점부터 조사를 시작한다. 우선 도착점인 경우에는 stack에 push한 후 탐색을 종료한다. 아닌 경우에 대해서는 우선 해당 노드의 right와 down의 참 여부를 확인한다. 각각의 경우에 대해 참이면서 방문하지 않았으면 해당 경로로 이동한다. 그 후 좌측과 위쪽에 대해서도 비슷한 방식으로 체크하고, 이동 시에는 stack에 push하고, 방문을 표시한다. 아닌 경우에 대해서는 stack에서 pop을 해줘 이전 노드로 돌아간다.

BFS는 구현하는 방식은 유사하지만 queue를 이용하는 특징이 있다. 이 경우에는 DFS와 구현은 유사하지만 DFS에서 stack의 push, pop의 부분에 대해서 enqueue와 dequeue로 바뀌어서 구현된다는 차이점이 있다.

.........................