13주차 예비보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20191612 이름: 윤기웅

**1. (문제 및 제목은 작성하지 말 것. 답안만 작성하면 됩니다.)**

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

먼저, 그래프는 노드(정점)와 간선(에지)으로 구성됩니다. 그래프를 탐색할 때 시간 복잡도는 주로 노드의 개수(V)와 간선의 개수(E)에 의존합니다. 이때, V는 그래프의 정점 수를 의미하고, E는 그래프의 간선 수를 의미합니다. DFS는 가능한 한 깊숙이 들어가면서 탐색하는 방법입니다. DFS는 스택 자료 구조를 이용해 구현할 수 있으며, 재귀를 이용해 구현할 수도 있습니다. DFS는 한 경로를 따라 계속 이동하다가 더 이상 갈 수 없으면 가장 최근에 방문했던 정점으로 되돌아가 다른 경로를 탐색합니다. DFS는 각 정점을 한 번 방문하고 간선도 각각 한 번 검사하기에 전체 시간복잡도는 O(V+E)가 됩니다.

BFS는 현재 노드의 모든 인접 노드를 먼저 탐색한 후, 그 다음 인접 노드를 탐색하는 방법입니다. BFS는 큐 자료 구조를 이용해 구현됩니다. BFS는 시작 정점에서 가까운 정점부터 탐색을 진행하며, 이를 통해 최단 경로를 찾을 때 유용합니다. BFS도 각 정점 한 번 방문하고 간선을 다 검사하기에 시간복잡도는 O(V+E)가 됩니다.

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

**2.**

12주차에서는 BFS와 DFS를 구현하기 위해 미리 정점을 만들어 인접 리스트를 사용하였습니다. 이번 주차에서는 인접 리스트 대신 방문 체크 배열과 백트래킹을 활용하여 재귀적으로 DFS를 구현하고, 큐를 사용하여 BFS를 구현하고자 합니다.

먼저, 가장 최단 경로를 기록하고 현재 경로를 추적할 벡터 구조체인 shortestRoute와 currentRoute, 탐색한 모든 경로를 저장할 allRoutes 벡터 구조체를 생성합니다. 미로의 정보를 저장한 maze 배열에서 벽은 0, 방은 1로 표시하여 탐색을 시작합니다. 방이고 아직 방문하지 않은 경우 방문 체크를 하고, 현재 위치를 currentRoute와 allRoutes에 추가합니다. 그 후 주변 방을 탐색하면서 해당 위치로 재귀를 시작합니다.

재귀를 진행하며 경로의 길이를 측정하기 위해 매 호출 시마다 경로의 수를 하나씩 증가시킵니다. 재귀가 끝나면 currentRoute에서 현재 위치를 pop하여 이전 위치로 돌아가 다시 탐색을 이어갑니다. 모든 경로를 탐색하기 위해 방문 배열을 초기화하여 재방문이 가능하도록 합니다. 탈출 위치에 도달하여 재귀가 종료될 때, 해당 경로가 지금까지의 가장 짧은 경로라면 shortestRoute에 currentRoute에 담긴 위치들을 기록합니다.

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

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