2025-1 데이터구조및프로그래밍실습 HW

학번_이름: 전자공학부 **202117395** 김동훈

문제 : DFS와 BFS의 장/단점 또는 특징은 어떤 것이 있을까요?

DFS

특징: DFS는 한 노드를 깊게 탐색한 후 더 이상 갈 곳이 없으면 되돌아오며 다음 노드를 탐색하는 방식입니다. 스택 또는 재귀 방식으로 구현되며, 현재 경로상의 노드들만 메모리에 저장하는 수직적 탐색 방법입니다. 백트래킹을 통해 이전 노드로 돌아가며 탐색을 계속하는 특성을 가집니다.

장점: DFS는 공간 복잡도가 O(h)(h는 최대 깊이)로 메모리 효율성이 뛰어나며, 재귀를 이용하면 구현이 간결하고 직관적입니다. 특정 경로의 존재 여부를 빠르게 판단할 수 있어 백트래킹 문제에 특히 적합하며, 모든 경우의 수를 체계적으로 탐색할 수 있습니다. 미로 탐색, N-Queen 문제, 스도쿠, 연결 요소 찾기, 사이클 탐지, 위상 정렬 등에 효과적으로 활용됩니다.

단점: DFS는 최단 거리를 보장하지 않으며, 먼저 발견한 경로가 최적 경로가 아닐 수 있습니다. 깊이가 너무 깊을 경우 스택 오버플로우가 발생할 수 있고, 특히 Python에서는 재귀 한계가 약 1000회로 제한되어 있어 주의가 필요합니다. 또한 사이클이 있는 그래프에서 방문 체크 없이는 무한 루프에 빠질 위험이 있으며, 목표가 얕은 곳에 있어도 깊게 탐색할 수 있어 비효율적일 수 있습니다.

BFS

특징: BFS는 시작 노드에서 가까운 노드부터 넓게 탐색하는 방식으로, 큐를 이용하여 레벨별로 탐색을 수행합니다. 각 레벨의 모든 노드를 완전히 탐색한 후 다음 레벨로 진행하는 수평적 탐색 방법이며, FIFO(First In First Out) 방식으로 노드를 처리합니다. 시작점으로부터의 거리가 가까운 순서대로 방문하는 특성을 가집니다.

장점: BFS는 가중치가 없는 그래프에서 시작점으로부터의 최단 경로를 보장하며, 각 노드를 한 번씩만 방문하므로 사이클이 있어도 안전하게 종료됩니다. 레벨별 탐색 특성으로 거리별 체계적인 분석이 가능하여 단계적 처리에 적합하며, 해가 존재한다면 반드시 찾을 수 있는 완전성을 보장합니다. 최단 거리 계산, 레벨 순서 탐색, 게임에서의 최소 이동 횟수 계산, 소셜 네트워크 분석, 이미지 처리의 Flood Fill 알고리즘 등에 효과적으로 활용됩니다.

단점: BFS는 각 레벨의 모든 노드를 큐에 저장해야 하므로 공간 복잡도가 O(w)(w는 최대 너비)로 메모리 사용량이 높습니다. 목표가 깊은 곳에 있을 때는 많은 중간 노드들을 탐색해야 하므로 비효율적일 수 있으며, 큐 자료구조 관리와 레벨 추적이 필요하여 구현이 상대적으로 복잡합니다. 또한 너비가 넓은 그래프에서는 큐에 매우 많은 노드가 쌓여 메모리 부족 문제가 발생할 수 있습니다.
