13주차 예비보고서

전공: 아트엔테크놀로지 학년: 3학년 학번: 20191172 이름: 함승우

DFS는 깊이 우선 탐색으로 기본 원리는 갈 수 있는 만큼 최대한 깊이 가고, 더 이상 갈 곳이 없으면 이전 정점으로 돌아가는 것이다. 2주차 실습에서 기존의 정보를 담은 adjacency matrix로 구현을 했고, DFS를 stack으로 구현하기 위해서는 discovered 배열을 사용해야 한다. discovered 배열이란 이미 방문한 노드들의 array를 저장했다.

Void DFS(int curr){

* stack을 생성한다., 시작 값을 stack에 삽입한다.
* 삽입한 값의 방문 flag를 true라고 한다.
* while() = stack이 빌 때까지 진행
* stack에서 값 하나를 추출한다.
* -----discovered array에서 이 값이 없으면 discovered에 삽입하고, 있으면 다시 같은 stack에 또 다른 값으로 계산한다.

})

DFS는 adjacency matrix로 구현을 했기에 정점 한 개 당 N번의 for loop를 돌아서 총 시간 복잡도는 O(N^2)에 해당한다.

BFS는 queue를 사용하여, 가로를 한 줄 씩 읽는다. 마찬가지로 정보를 adjacency matrix로 구현을 했다, 또한 위 DFS와 마찬가지로 이미 방문한 노드들을 저장하는 discovered array를 구성했다.

Void BFS(int curr){

* queue를 생성한다.
* queue에 현재 값을 push 한다
* push 한 값의 방문 flag를 true라고 한다.
* while() = queue가 빌 때까지 진행
* queue에서 추출
* discovered에 있으면 또 다른 queue의 값으로 연산하고, 없으면 삽입한다.

})

정점 한 개당 N번의 for loop을 돌기 때문에 하나의 값에 대한 시간 복잡도는 O(N)이지만, 이를 모든 Node에 실시하게 되면 총 시간 복잡도는 O(N^2)이다.