2020년도 2학기 컴퓨터공학설계및실험I

14주차 예비 보고서

20170175 김태안

1. 실습 목적

본 실험에서는 2주차의 실습에서 만든 OpenFrameWork 응용 프로그램을 수정하여 DFS 방법 (실습)과 BFS 방법 (과제)으로 미로를 탐색하여 그 결과를 화면에 표시하는 프로그램을 작성한다.

1. 관련 이론

DFS는 출발 정점 v를 방문하며 시작한다. 다음으로 인접한 정점을 인접 리스트에 저장한 뒤, v에 인접하면서 아직 방문하지 않은 정점 w를 선택해 w에 인접하면서 아직 방문하지 않은 새 정점을 선택하며 DFS를 다시 시작한다. 이를 재귀적으로 반복해 모든 정점을 탐색한다.[1] 따라서 DFS를 통해 미로를 탐색할 경우. 시간 복잡도는 모든 칸의 탐색 시간인 O(HIEGHT\*WIDTH)이다.

DFS는 현재 탐색 중인 정점을 스택에 넣어 관리한다. 이때, 최악의 경우, 모든 정점이 연속해 이어져 있어 모든 정점을 스택에 삽입해야 한다. 따라서 미로의 경우, 칸의 개수만큼 공간이 필요하며, 이때의 공간 복잡도는 O(WIDTH\*HIEGHT)이다.

BFS는 정점 v에서 시작하여 방문 표시를 한 뒤, v의 인접 정점을 모두 방문한다. 그 다음, v의 첫 인접 정점에 인접한 정점부터 모든 인접 정점의 인접 정점을 방문한다.[1] 이를 반복해 모든 정점을 탐색한다. DFS와 마찬가지로 모든 정점을 탐색하므로, 미로 탐색에서의 시간 복잡도는 O(HEIGHT\*WIDTH)이다.

BFS는 다음에 탐색할 정점을 큐에 저장해 불러온다. 이때, 최악의 경우, 시작과 모든 정점이 인접해 모든 정점을 큐에 저장해야 한다. 미로의 경우, 칸의 개수만큼 공간이 필요하므로 공간 복잡도는 O(WIDTH \* HIEGHT)이다.

1. 실습 방법

2주차 실습에서 미로의 정보를 담은 인접 리스트를 생성하였다. 인접 리스트를 활용하여 DFS와 BFS를 구현한다. 인접 리스트는 각 칸을 0부터 HEIGHT\*WIDTH의 번호를 가졌다고 가정한다. 따라서 탐색을 하며 HEIGHT\*WIDTH 칸을 찾으면 알고리즘을 종료하고 탐색 결과를 반환한다.

DFS의 수도코드는 다음과 같다.

|  |
| --- |
| void dfs (int v) {  현재 칸을 탐색했다고 표시  스택에 저장  if (v = WIDTH\*HEIGHT)  found = 1  if (found == 1)  return  for (w = 현재 칸에 연결된 칸)  if (방문한 적이 없으면)  dfs(w);  스택에서 제거  } |

현재 연결된 칸은 연결 리스트를 탐색하며 찾는다. 탐색한 칸은 visited 배열을 생성해 탐색 기록을 저장한다. HEIGHT\*WIDTH 칸을 찾으면 알고리즘을 종료한다. 이때 탐색한 경로는 스택에 저장된 칸을 이은 결과이다.

BFS의 수도코드는 다음과 같다.

|  |
| --- |
| void bfs (int v) {  현재 칸을 탐색했다고 표시  큐에 추가  while (큐에 값이 있다면) {  v = 큐의 맨 처음  큐의 맨 처음 값 삭제  for (w = v에 연결된 칸)  if (w를 탐색한 적이 없으면) {  큐에 추가  탐색했다고 표시  parent 저장  if (w == HEIGHT\*WIDTH)  종료  }  }  } |

parent 배열은 탐색에서의 진입 정점을 저장한다. 탐색을 종료한 뒤, HEIGHT\*WIDTH부터 parent를 역추적하면 경로를 구할 수 있다.

1. 기타

**인접 행렬을 사용한 DFS. BFS의 시간 복잡도**

실습에서는 인접 리스트를 통해 그래프를 표현하였으나, 인접 행렬을 통해서도 그래프를 표현할 수 있다. 이때, DFS와 BFS의 알고리즘은 같으나 시간 복잡도가 달라지게 된다. DFS의 경우, 인접 행렬을 탐색해 v에 인접한 정점을 찾는데 걸리는 시간은 O(n)이다. 따라서 최악의 경우, n에 대해 n번의 탐색을 해야 하므로 O(n2)의 시간이 걸린다. BFS의 경우, 방문되는 각 정점에 대해 while 루프는 O(n)의 시간이 소요된다. 따라서 전체 시간은 O(n2)의 시간이 소요된다.[1]

1. 참고 문헌

[1] Ellis Horowitz et al., “6. Graph.” *Fundamentals of Data Structures in C*, University Press, 2008.