11주차 예비보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20231515 이름: 김다은

**1. 문제 해결에서 언급한 미로 생성 알고리즘들에서 Eller's algorithm을 제외한 나머지 알고 리즘 중 하나를 선택하여 이를 조사하고 이해한 후 그 방법을 기술하시오.**

프림 알고리즘을 이용해 미로를 만드는 것이 가능하다. 프림의 알고리즘은 최소비용신장트리(Minimum Spanning Tree)를 구성하는 알고리즘이다. 그래프의 모든 노드를 포함하고 각 노드를 최소 비용으로 연결하는 트리를 만들어 미로 생성에 사용한다.

작동 절차는 다음과 같다.

1. 초기화
   1. 미로의 벽을 모두 세팅한다.
   2. 시작점을 선정하고 방문 처리한다.
2. 벽 선택
   1. 현재 칸에 인접한 벽들 중 1개를 랜덤으로 선택한다.
   2. 선택한 벽의 반대편의 칸을 확인한다.
3. 벽 허물기
   1. 만약 반대편 칸이 아직 방문되지 않았다면, 벽을 허물고 해당 칸을 방문 처리한다.
   2. 만약 이미 방문된 칸이라면, 벽을 그대로 둔다.
4. 모든 칸이 방문 처리될 때까지 2, 3번 과정을 반복한다.

**2. 본 실험에서 완전 미로(Perfect maze)를 만들기 위하여 선택한 알고리즘 구현에 필요한 자료구조를 설계하고 기술하시오. 설계한 자료구조를 사용하였을 경우 선택한 알고리즘의 시간 및 공간 복잡도를 보이시오.**

본 실험에서는 Eller’s Algorithm을 선택하였다. Eller’s Algorithm은 다음과 같은 절차로 진행된다.

1. 초기화
   1. 미로의 벽을 세팅한다.
2. 첫번째 row의 각 칸에 1~row까지의 숫자를 저장한다.
3. 기준 칸과 오른쪽 칸에 저장된 숫자가 다른 경우 랜덤으로 벽을 제거하거나 제거하지 않는다. 만약 제거한다면 오른쪽 칸과 이어진 모든 미로 칸의 숫자를 기준 칸의 숫자로 변경한다.
4. 기준 칸의 아래 벽을 랜덤으로 제거하거나 제거하지 않는다. 이때 각 숫자마다 적어도 1개의 아래 벽은 제거해야 한다.
5. 3번과 4번 과정을 마지막 row에 도달할 때까지 반복한다.
6. 마지막 열에서 오른쪽 벽을 지우는 연산을 진행한다. 이때 기준 벽과 오른쪽 벽에 저장된 숫자가 다른 경우에만 제거한다. 이후 오른쪽 칸과 이어진 미로에 기준 칸의 숫자를 저장한다.

문제를 해결하기 위해 다음과 같은 자료구조를 생각해볼 수 있다.

int dx[] = {-1, 0, 0, 1};

int dy[] = {0, -1, 1, 0};

미로에서 벽을 제거함으로써 이어진 두 길을 같은 숫자로 통일시켜야 한다. 이 과정에서 기준 칸에서 4방면(위, 아래, 왼쪽, 오른쪽)으로 탐색해야 한다. dx[], dy[]를 이용해 다음과 같이 나타내면 4방면을 탐색할 수 있다.

for(int k=0; k<4; k++) {

        int nx = x+dx[k];

        int ny = y+dy[k];

~

}

이를 응용해 searchmaze 함수를 구현할 수 있을 것이다. 여기서 board[x][y] = dst(기준 칸의 숫자)는 방문 처리의 역할도 동시에 수행한다. 이를 바탕으로 DFS를 구현한 모습이다.

void searchmaze(int x, int y, int target, int dst) {

    board[x][y] = dst;

    for(int k=0; k<4; k++) {

        int nx = x+dx[k];

        int ny = y+dy[k];

        if(0<=nx && nx<m && 0<=ny && ny<n && board[nx][ny] == target) {

            searchmaze(nx, ny, target, dst);

        }

    }

}

<시간복잡도>

미로의 행의 길이를 n, 열의 길이를 m이라고 생각하겠다. 각 열마다 행을 2번씩 탐색하게 된다. 따라서 O(n\*(2\*m)), O(n\*m)이라고 할 수 있다.

<공간복잡도>

미로의 정보를 2차원 배열에 저장하게 되므로 O(n\*m)으로 표현할 수 있다.