11주차 예비보고서

전공 : 경영학과 학년 : 4학년 학번 : 20190808 이름 : 방지혁

1. 문제 해결에서 언급한 미로 생성 알고리즘 들에서 Eller's algorithm을 제외한 나머지 알고리즘 중 하나를 선택하여 이를 조사하고 이해한 후 그 방법을 기술하시오.

Kruskal algorithm, prim's algorithm, recursing back tracking등의 방법이 있을 수 있다. 이 중 Kruskal algorithm에 대해 서술하고자 한다. 기존의 Kruskal algorithm은 minimum spanning tree, 즉 사이클이 없는 tree를 만드는 것이 주목적이다. 모든 cell을 각각의 node로 보고, 인접한 셀 사이의 벽을 edge로 봅니다. 이후, 간선들을 랜덤하게 선택하면서, 사이클이 생기지 않도록 벽을 제거한다. 더 자세히 서술하자면 N * M 개의 cell이 있고, 각 cell은 독립된 집합으로 본다. 그리고 모든 가능한 벽의 리스트를 만들고, 랜덤하게 섞는다. 해당 리스트에서 하나씩 꺼내면서 만약 두 셀이 서로 다른 집합에 속하면, 벽을 제거해 둘을 합치고, 같은 집합이면 벽을 유지해서 사이클을 방지한다. 그 결과 우리가 목표로 하는 '두 지점을 연결하는 경로가 오직 하나 존재하는 미로'인 완전미로를 완성할 수 있다.

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

총 4종류의 자료구조가 필요할 듯하다. 첫 번째는 바로 집합 배열로 1차원 배열이다. 현재 행에서 각 cell이 속한 집합 번호를 저장하기 위한 자료구조이다. 두 번째는 세로 벽을 나타내는 2차원 배열이다. 각 셀의 오른쪽 벽이 있는 지 여부를 저장한다. 예를 들어, walls[row][col]은 (row, col) cell과 (row, col + 1) cell 사이의 벽 존재 유무를 0과 1로 나타낸다. 이는 M * N 크기이다. 그리고 가로 벽 배열이 필요한데 이는 각 cell의 아래쪽 벽이 존재하는지 저장하는 것으로 M * N 크기이다. 예를 들어, walls[row][col]은 (row, col) cell과 (row + 1, col) cell 사이의 벽 존재 유무를 0과 1로 나타낸다. 마지막으로 집합이 연결되어 있는지 체크하는 배열이 필요하다. 이는 완전 미로 조건을

충족시키기 위해서 필요한 것으로, 각 set(집합)이 아래 행의 cell들 중 최소 하나 이상 연결되었는지 추적하기 위한 것이다. 이는 N 크기이다. 해당 알고리즘의 시간 복잡도의 각 행을 처리하는데 외부 루프에서 O(M)이 필요하다. 각 행에 대해서 집합체크 및 연결 처리에서 최대 O(N^2)의 시간 복잡도가 소요되기 때문에 총 O(M*N^2)의 시간 복잡도가 예상된다. 그리고 공간 복잡도의 경우 앞서 말했듯 할당하는 자료구조 중 가장 큰 것의 크기로 O(MN)이다.