8/20 - 2

최백준 choi@startlink.io

연구소

https://www.acmicpc.net/problem/14502

- N×M 크기의 직사각형 지도가 있고, 1×1크기의 칸으로 나누어져 있다. ($3 \le N, M \le 8$)
- 칸: 빈 칸, 벽
- 일부 빈 칸에는 바이러스가 있고, 인접한 빈 칸으로 계속해서 퍼져 나간다.
- 벽을 3개 세워서 바이러스가 퍼질 수 없는 곳의 크기를 구하는 문제

연구소

https://www.acmicpc.net/problem/14502

- N×M 크기의 직사각형 지도가 있고, 1×1크기의 칸으로 나누어져 있다. ($3 \le N, M \le 8$)
- 칸: 빈 칸, 벽
- 일부 빈 칸에는 바이러스가 있고, 인접한 빈 칸으로 계속해서 퍼져 나간다.
- 벽을 세운다는 내용을 잠시 제외하면
- 입력으로 주어진 상태에서 바이러스가 퍼질 수 없는 영역의 크기는 BFS로 구할 수 있다.

연구소

https://www.acmicpc.net/problem/14502

- N×M 크기의 직사각형 지도가 있고, 1×1크기의 칸으로 나누어져 있다. (3 ≤ N, M ≤ 8)
- 칸: 빈 칸, 벽
- 일부 빈 칸에는 바이러스가 있고, 인접한 빈 칸으로 계속해서 퍼져 나간다.
- 벽을 세운다는 내용을 잠시 제외하면
- 입력으로 주어진 상태에서 바이러스가 퍼질 수 없는 영역의 크기는 BFS로 구할 수 있다.
- 칸은 정점, 인접한 칸의 관계는 간선으로 나타내면, 바이러스에서 시작해서 연결된 모든 정점을 방문하는 문제가 되어버리기 때문.
- 시간 복잡도: O(NM)

연구소

https://www.acmicpc.net/problem/14502

- N×M 크기의 직사각형 지도가 있고, 1×1크기의 칸으로 나누어져 있다. ($3 \le N, M \le 8$)
- 칸: 빈 칸, 벽
- 일부 빈 칸에는 바이러스가 있고, 인접한 빈 칸으로 계속해서 퍼져 나간다.
- 벽을 3개 세우는 경우의 수: $(NM)^3$
- 벽을 세운 다음 안전 영역의 크기를 구하는 방법: BFS 또는 DFS, $O(NM)$
- 총 $O((NM)^4)$가 나오는데, $N, M \le 8$이기 때문에, 시간 안에 해결할 수 있다.

연구소

https://www.acmicpc.net/problem/14502

- BFS 소스: http://codeplus.codes/24a2143fe0bf4d84b9f10a002e797178
- DFS 소스: http://codeplus.codes/1bda5371d7224d608db68231c96f5687

벽 부수고 이동하기

https://www.acmicpc.net/problem/2206

- NxM의 행렬로 나타내는 지도에서 (1, 1)에서 (N,M)으로 최단 거리로 이동하는 문제
- 0은 빈 칸, 1은 벽
- 단, 벽은 한 번 부수고 지나갈 수 있다

벽 부수고 이동하기

https://www.acmicpc.net/problem/2206

- 벽을 부순다는 조건이 없으면 일반적인 미로 탐색 문제이다
- 어떤 칸에 방문했을 때, 벽을 부순 적이 있는 경우와 아직 부순 적이 없는 경우는 다른 경우 이기 때문에
- 상태 (i, j) 대신에 (i, j, k) (k == 0이면 벽을 부순 적이 없음, 1이면 있음) 으로 BFS 탐색을 진행한다.

벽 부수고 이동하기

https://www.acmicpc.net/problem/2206

- 소스: http://codeplus.codes/e59c030ff8b940b682ba71ab865730d0

연구소 3

https://www.acmicpc.net/problem/17142

- 크기가 N×N인 연구소가 있고, 빈 칸, 벽, 바이러스이다.
- 바이러스는 활성/비활성 상태가 있다.
- 가장 처음에 모든 바이러스는 비활성 상태이고, 바이러스 M개를 활성 상태로 바꾸려고 한다.
- 활성 바이러스는 상하좌우로 인접한 모든 빈 칸으로 동시에 복제되고, 1초가 걸린다.
- 비활성 바이러스가 있는 곳으로 활성 바이러스가 이동하면 비활성이 활성으로 변한다.
- 모든 빈 칸에 바이러스를 퍼뜨리는 최소 시간을 구하는 문제

연구소 3

https://www.acmicpc.net/problem/17142

```
2 0 0 0 1 1 0
0 0 1 0 1 2 0
0 1 1 0 1 0 0
0 1 0 0 0 0 0
0 0 0 2 0 1 1
0 1 0 0 0 0 0
2 1 0 0 0 0 2
```

- 0: 빈 칸, 1: 벽, 2: 비활성 바이러스

연구소 3

https://www.acmicpc.net/problem/17142

```
* 6 5 4 - - 2
5 6 - 3 - 0 1
4 - - 2 - 1 2
3 - 2 1 2 2 3
2 2 1 0 1 - -
1 - 2 1 2 3 4
0 - 3 2 3 4 *
```

- M = 3인 경우 방법 하나
- 활성 바이러스: 0, 비활성 바이러스: *, 벽: -, 정수: 바이러스가 퍼지는 시간

연구소 3

https://www.acmicpc.net/problem/17142

```
0 1 2 3 - - 2
1 2 - 3 - 0 1
2 - - 2 - 1 2
3 - 2 1 2 2 3
3 2 1 0 1 - -
4 - 2 1 2 3 4
* - 3 2 3 4 *
```

- M = 3인 경우 최소 시간이 걸리는 방법
- 활성 바이러스: 0, 비활성 바이러스: *, 벽: -, 정수: 바이러스가 퍼지는 시간

연구소 3

https://www.acmicpc.net/problem/17142

- N ≤ 50
- M ≤ 10
- M ≤ 비활성 바이러스의 수 ≤ 10

연구소 3

https://www.acmicpc.net/problem/17142

- $N \leq 50$
- $M \leq 10$
- $M \leq$ 비활성 바이러스의 수 ≤ 10
- 바이러스를 선택하고, 모든 칸으로 퍼뜨리는 시간을 계산해본다.
- 바이러스를 선택할 수 있는 방법의 수: 2^{10}
- 모든 칸으로 퍼뜨리는 시간의 계산: BFS를 이용해 $O(N^2)$

연구소 3

https://www.acmicpc.net/problem/17142

- BFS에서는 바이러스가 빈 칸과 같은 의미를 갖지만, 정답을 구할 때는 아니다.
- 정답을 구할 때는 빈 칸까지 가는 거리의 최댓값만 구해야 한다.

연구소 3

https://www.acmicpc.net/problem/17142

- 소스: http://codeplus.codes/b2d4eed7aff24fb0adccdcf8489ffe2d

움직이는 미로 탈출

https://www.acmicpc.net/problem/16954

- 크기가 8×8인 체스판이 있고, 모든 칸에는 빈 칸 또는 벽이다.
- 가장 왼쪽 아랫 칸에서 가장 오른쪽 윗 칸으로 이동할 수 있는지 없는지 구하는 문제
- 벽은 1초에 한 칸씩 아래로 내려온다.
- 벽이 있는 칸으로 이동할 수 없고, 이동한 칸에 벽이 내려오면 더 이상 이동할 수 없다.

움직이는 미로 탈출

https://www.acmicpc.net/problem/16954

- 8초가 지나면 벽이 없어진다.

움직이는 미로 탈출

https://www.acmicpc.net/problem/16954

- 지도를 총 9개 준비해서, 0초 후, 1초 후, 2초 후, …, 8초 후를 만들고 BFS를 수행할 수 있다.
- (r, c, t): t초 후에 (r, c)에 있을 때 최소 시간
- 8초 후부터는 t를 증가시키는 의미가 없다.

움직이는 미로 탈출

https://www.acmicpc.net/problem/16954

- 실제로는 지도를 9개나 만들 필요는 없다.
- 특정 시점이 t초 후에 벽이 있는지 없는지는 알아낼 수 있기 때문이다.
- t초 후에 (r, c)로 벽이 내려왔다면, 그 벽은 (r-t, c)에 있던 벽이다.

움직이는 미로 탈출

https://www.acmicpc.net/problem/16954

- 소스: http://codeplus.codes/1fba1df3ce894abaac4061781fbfb7dc