파이썬으로 배우는 알고리즘 기초

Chap 5. 되추적(백트래킹)

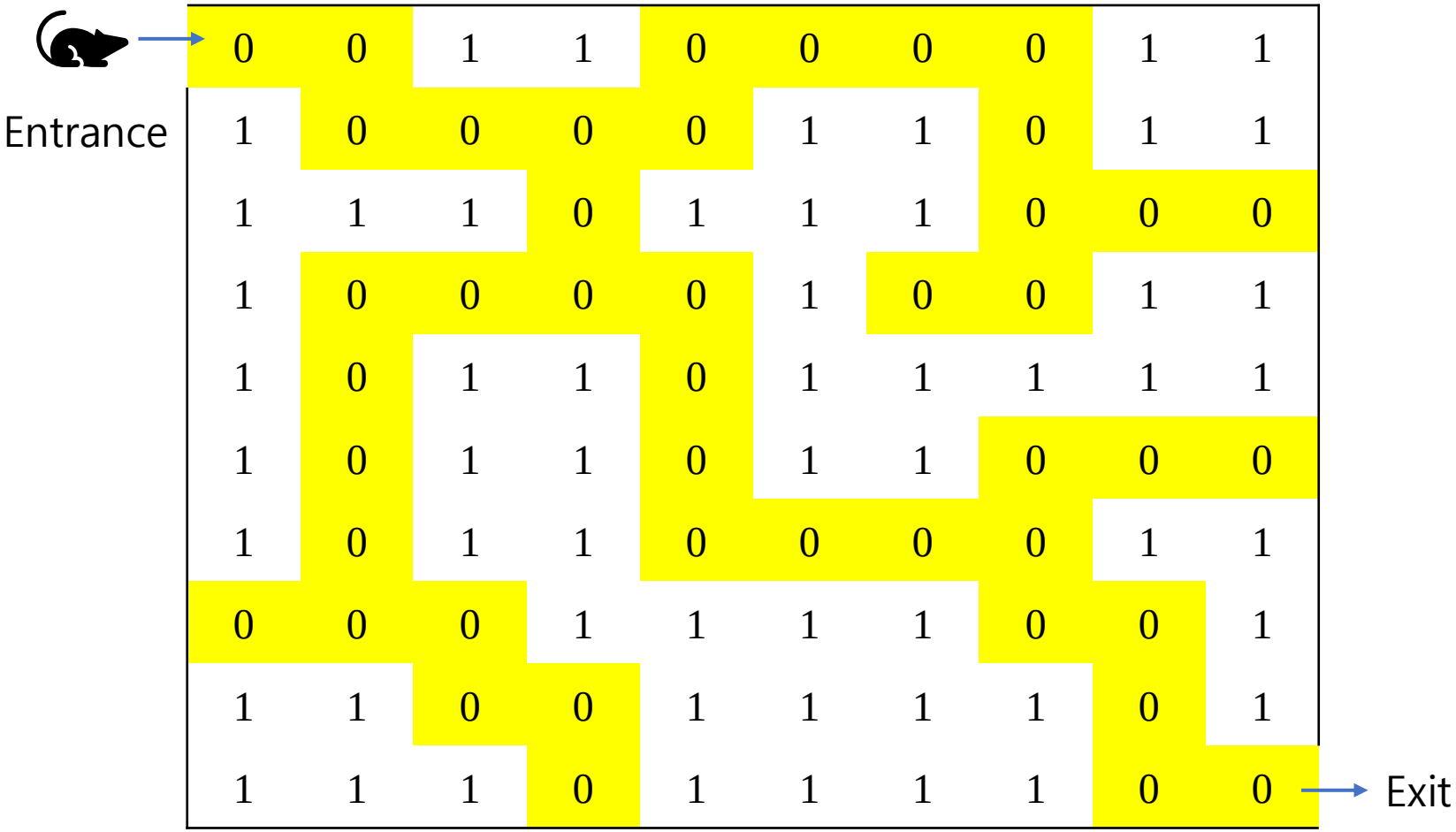
5.1 백트래킹과 n-Queens 문제

5.1 백트래킹과 n-Queens 문제

- **되추적** (backtracking)
  - **임의의 집합**(set)**에서 주어진 기준**(criterion)**대로**
    - **원소의 순서**(sequence)**를 선택하는 문제를 푸는 데 적합**
  - **트리 자료구조의 변형된 깊이우선탐색**(DFS: depth-first-search)
  - **모든 문제 사례에 대해서 효율적이지 않지만,**
    - **많은 문제 사례에 대해서 효율적이다.**
    - **예)** n-Queens, **부분집합의 합**, 0-1 **배낭문제**, etc.

5.1 백트래킹과 n-Queens 문제

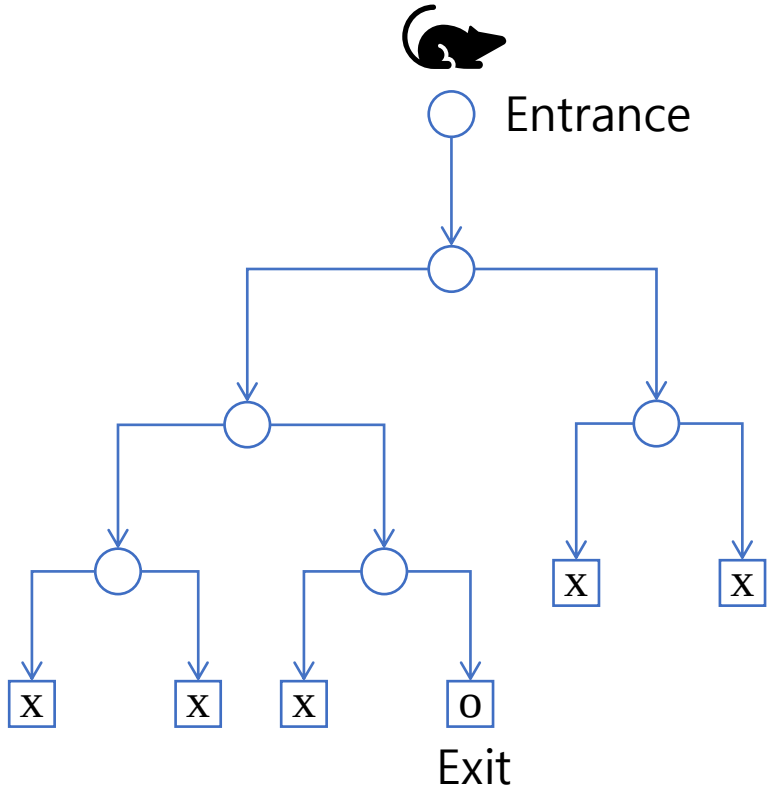
- **미로찾기** 문제



5.1 백트래킹과 n-Queens 문제

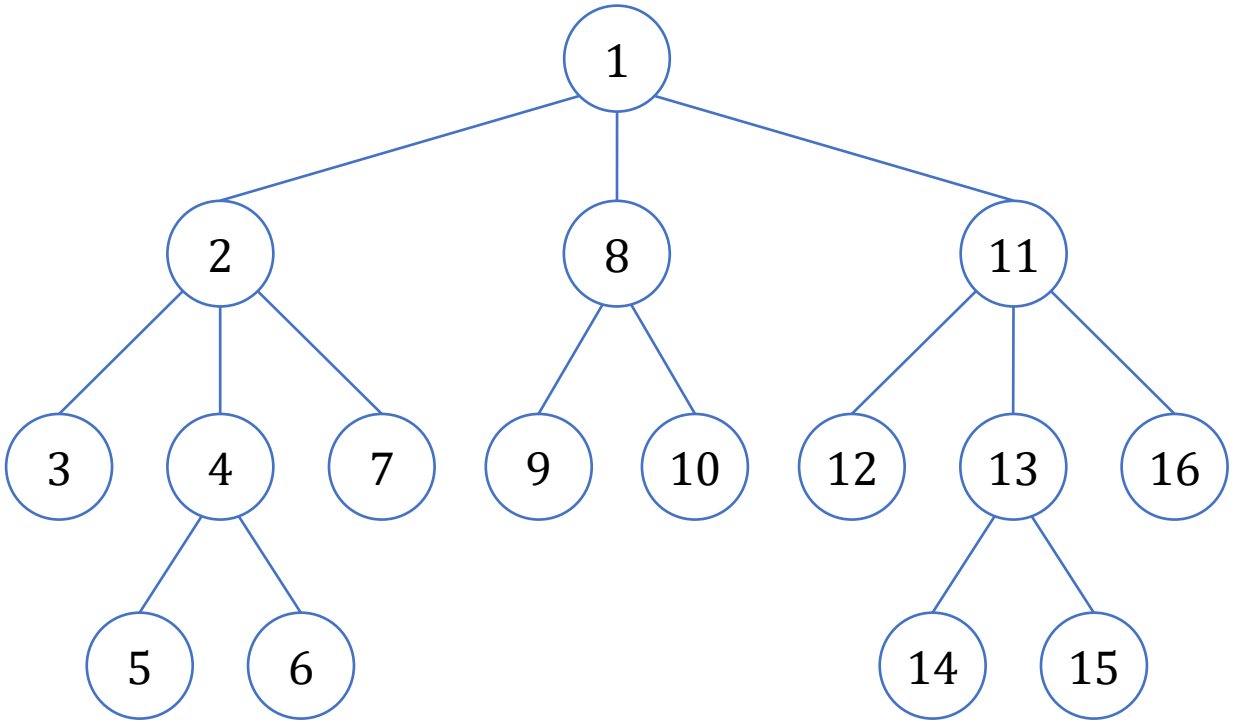
- **미로찾기 문제를 트리 탐색 문제로 해석**
  - DFS를 통한 문제 해결: 트리 구조의 preorder 방문

0	0	1	1	0	0	0	0	1	1
1	0	0	0	0	1	1	0	1	1
1	1	1	0	1	1	1	0	0	0
1	0	0	0	0	1	0	0	1	1
1	0	1	1	0	1	1	1	1	1
1	0	1	1	0	1	1	0	0	0
1	0	1	1	0	0	0	0	1	1
0	0	0	1	1	1	1	0	0	1
1	1	0	0	1	1	1	1	0	1
1	1	1	0	1	1	1	1	0	0



5.1 백트래킹과 n-Queens 문제

- **상태공간트리** (State Space Tree)
  - 상태 공간: 해답을 탐색하기 위한 탐색 공간
  - 상태공간트리: 탐색 공간을 트리 형태의 구조로 암묵적으로 해석



5.1 백트래킹과 n-Queens 문제

- **백트래킹** 기법
  - 상태공간트리를 깊이우선탐색으로 탐색
  - 방문 중인 노드에서 더 하위 노드로 가면 해답이 없을 경우
    - 해당 노드의 하위 트리를 방문하지 않고 부모 노드로 되돌아 감 (backtrack)

- **유망함** (promising)
  - 방문 중인 노드에서 하위 노드가 해답을 발견할 가능성이 있으면 유망(promising)
  - 하위 노드에서 해답을 발견할 가능성이 없으면 유망하지 않음(nonpromising)

5.1 백트래킹과 n-Queens 문제

- 백트래킹과 가지치기 (pruning)
  - 백트래킹: 상태공간트리를 DFS로 탐색
  - 방문 중인 노드가 유망한지 체크
  - 만약 유망하지 않으면, 부모 노드로 되돌아감 (backtrack)

- 가지치기 (pruning)
  - 유망하지 않으면 하위 트리를 가지치기함
  - 가지치기한 상태: 방문한 노드의 방문하지 않는 하위 트리 (pruned state)

5.1 백트래킹과 n-Queens 문제

- 일반적인 백트래킹 알고리즘

```
void checknode (node v)
{
  node u;
  if (promising(v))
    if (v에 해답이 있으면)
      해답을 출력;
    else
      for (v의 모든 자식 노드 u에 대해서)
        checknode(u);
}
```

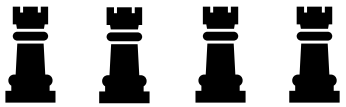
promising function

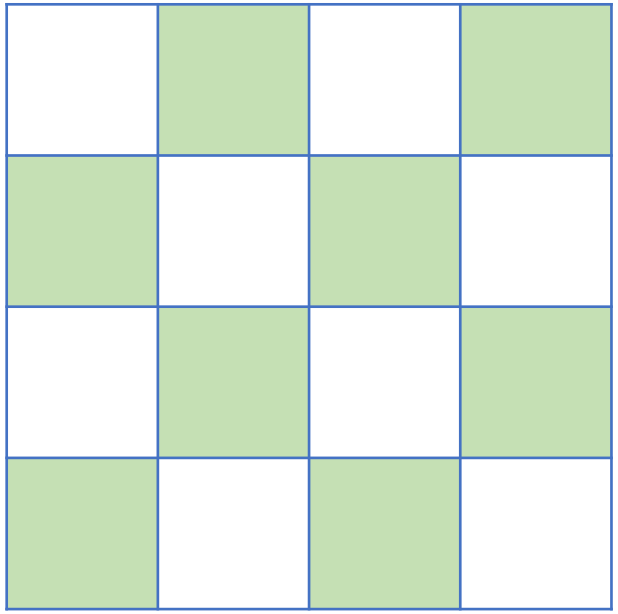
5.1 백트래킹과 n-Queens 문제

- 백트래킹 알고리즘의 구현
  - 상태공간트리를 실제로 구현할 필요는 없음
  - 현재 조사중인 가지의 값에 대해 추적만 하면 됨
  - 상태공간트리는 암묵적으로 존재한다고 이해하면 됨

5.1 백트래킹과 n-Queens 문제

- n-Queens 문제
  - 8-Queens ($n = 8$) 문제의 일반화된 문제
  - $n \times n$ 체스보드에 n개의 퀸을 배치하는 문제
    - 어떤 퀸도 다른 퀸에 의해서 잡아먹히지 않도록 배치해야 함
    - 즉, 같은 행, 열, 대각선에는 다른 퀸을 놓을 수 없음



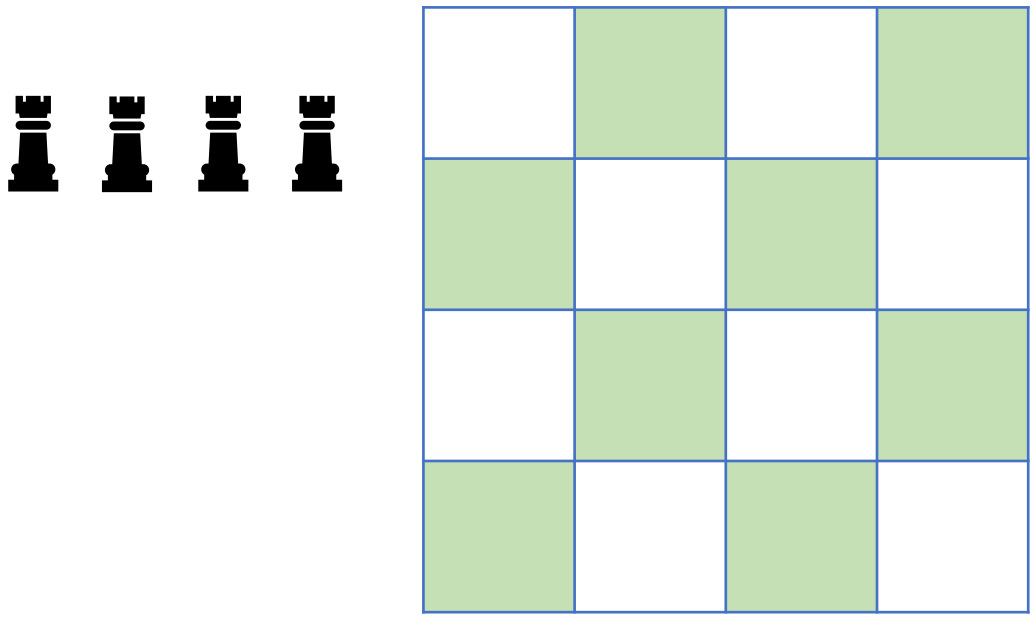


5.1 백트래킹과 n-Queens 문제

- n-Queens 문제: 백트래킹 (backtracking)
  - 백트래킹으로 문제 해결:
    - 임의의 집합에서 기준에 따라 원소의 순서를 선택
  - n-Queens 문제에 적용:
    - 임의의 집합(set): 체스보드에 있는 n^2개의 가능한 위치
    - 기준(criterion): 새로 놓을 퀸이 다른 퀸을 위협할 수 없음
    - 원소의 순서(sequence): 퀸을 놓을 수 있는 n개의 위치

5.1 백트래킹과 n-Queens 문제

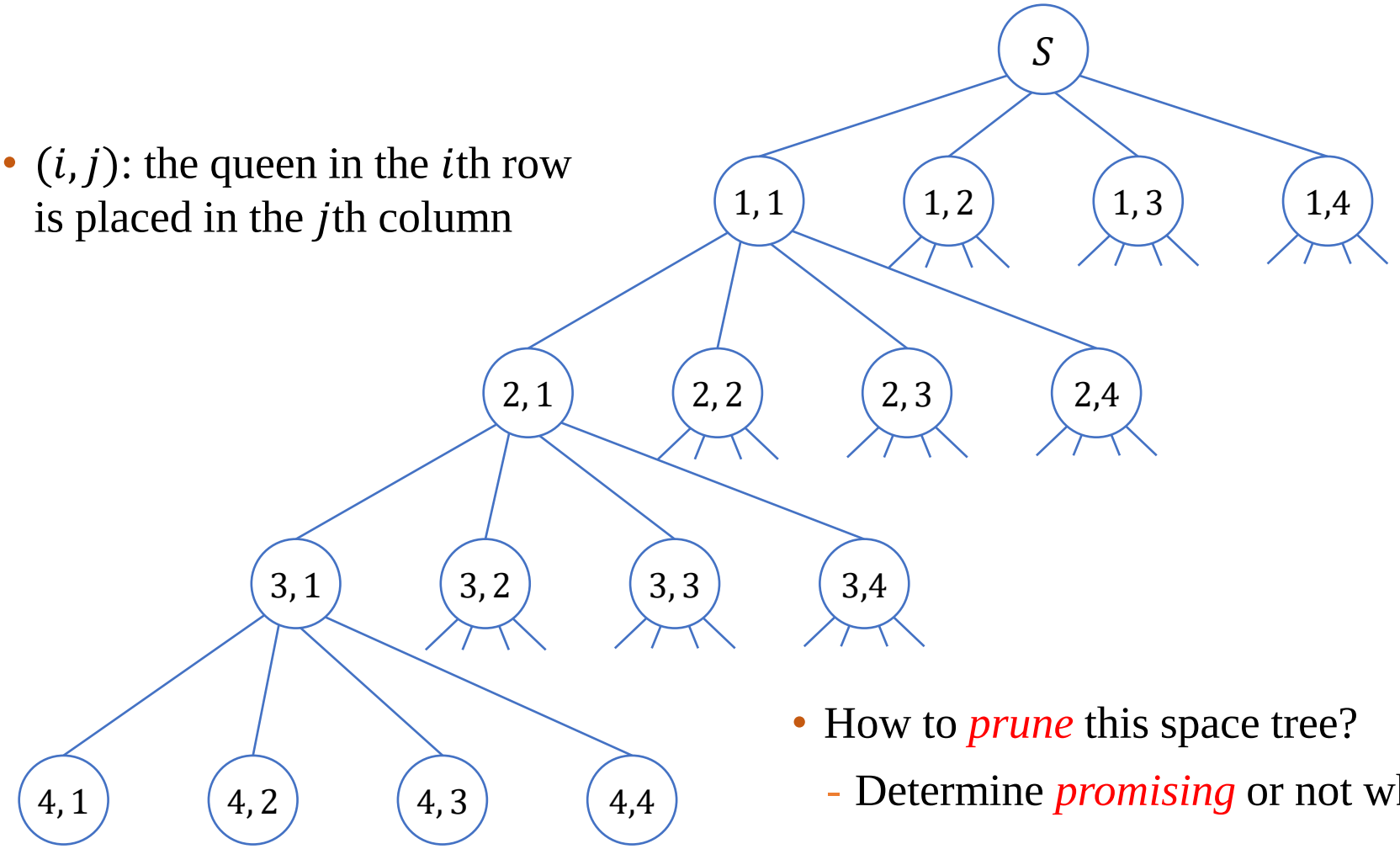
- 4-Queens 문제 ($n = 4$)
  - 4개의 퀸을 4×4 체스보드에 배치
    - 일단, 기본 가정으로 같은 행(row)에는 놓을 수 없음
  - 후보 해답: $4 \times 4 \times 4 \times 4 = 256$ 가지의 탐색 공간이 있음



5.1 백트래킹과 n-Queens 문제

■ 상태공간트리의 구성

- (i, j): the queen in the ith row is placed in the jth column



- How to *prune* this space tree?
  - Determine *promising* or not while traversing the tree

주니온TV@Youtube

자세히 보면 유익한 코딩 채널

https://bit.ly/2JXXGqz

- 여러분의 **구독**과 **좋아요**는 강의제작에 큰 힘이 됩니다.
- 강의자료 및 소스코드: **구글 드라이브**에서 다운로드
  (다운로드 주소는 영상 하단 설명란 참고)

  https://bit.ly/3fN0q8t