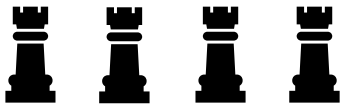
파이썬으로 배우는 알고리즘 기초

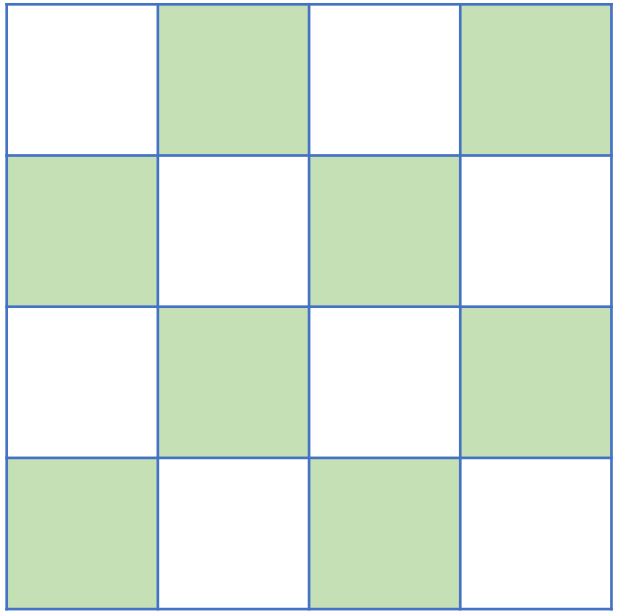
Chap 5. 되추적(백트래킹)

5.2 n-Queens 문제의 구현

5.2 n-Queens 문제의 구현

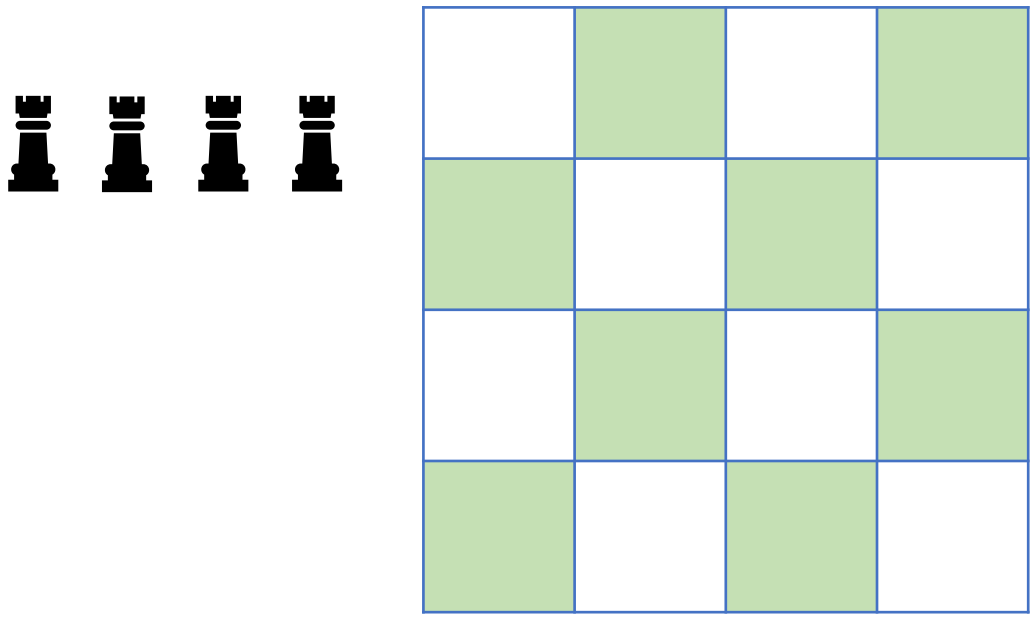
- n-Queens 문제
  - $n \times n$ 체스보드에 n개의 퀸을 배치하는 문제
    - 어떤 퀸도 다른 퀸에 의해서 잡아먹히지 않도록 배치해야 함
    - 즉, 같은 행, 열, 대각선에는 다른 퀸을 놓을 수 없음





5.2 n-Queens 문제의 구현

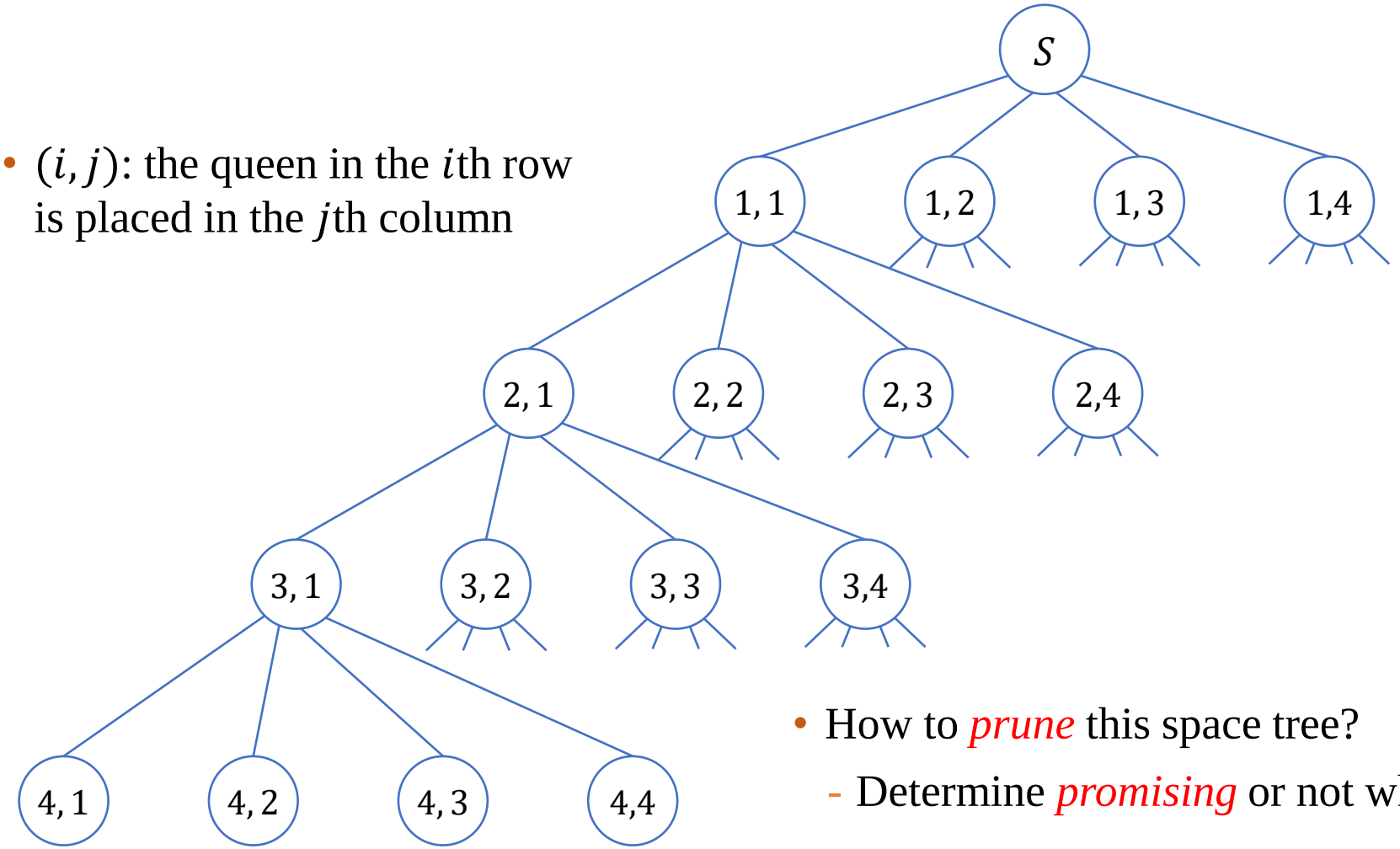
- 4-Queens 문제 ($n = 4$)
  - 4개의 퀸을 4×4 체스보드에 배치
    - 일단, 기본 가정으로 같은 행(row)에는 놓을 수 없음
  - 후보 해답: $4 \times 4 \times 4 \times 4 = 256$ 가지의 탐색 공간이 있음



5.2 n-Queens 문제의 구현

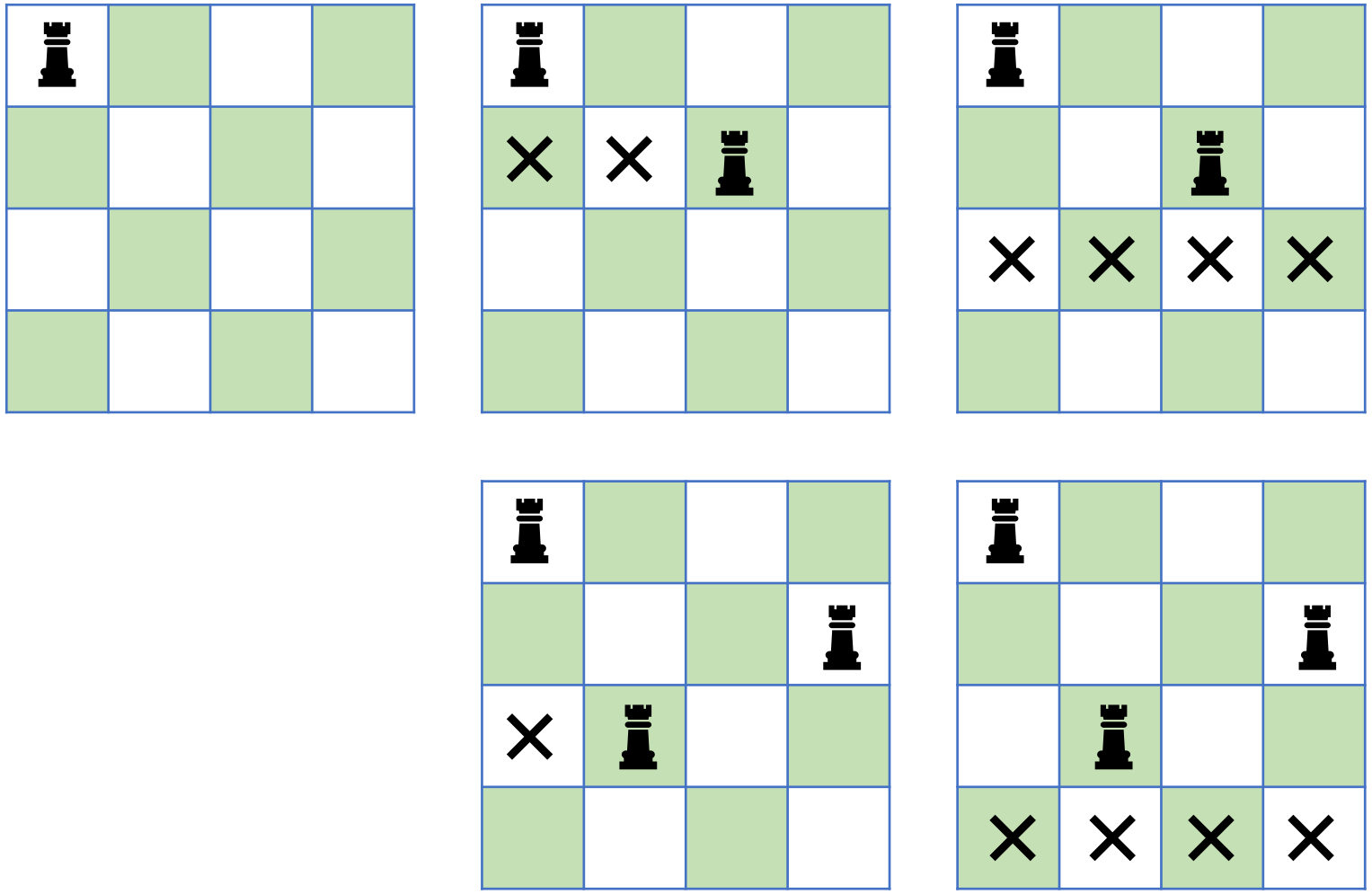
상태공간트리의 구성

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

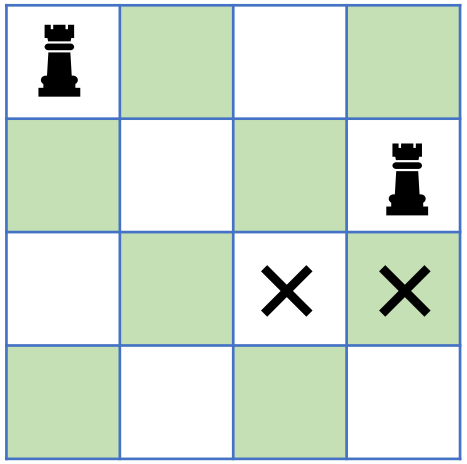


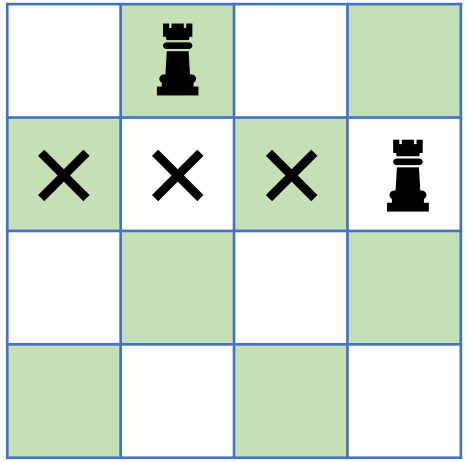
- How to *prune* this space tree?
  - Determine *promising* or not while traversing the tree

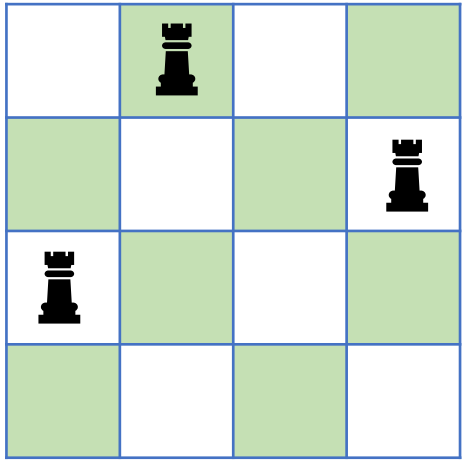
5.2 n-Queens 문제의 구현

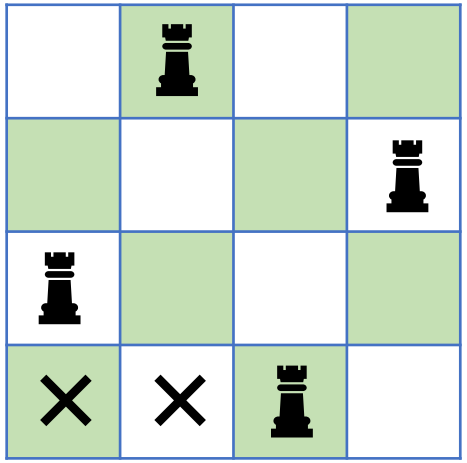


5.2 n-Queens 문제의 구현









5.2 n-Queens 문제의 구현

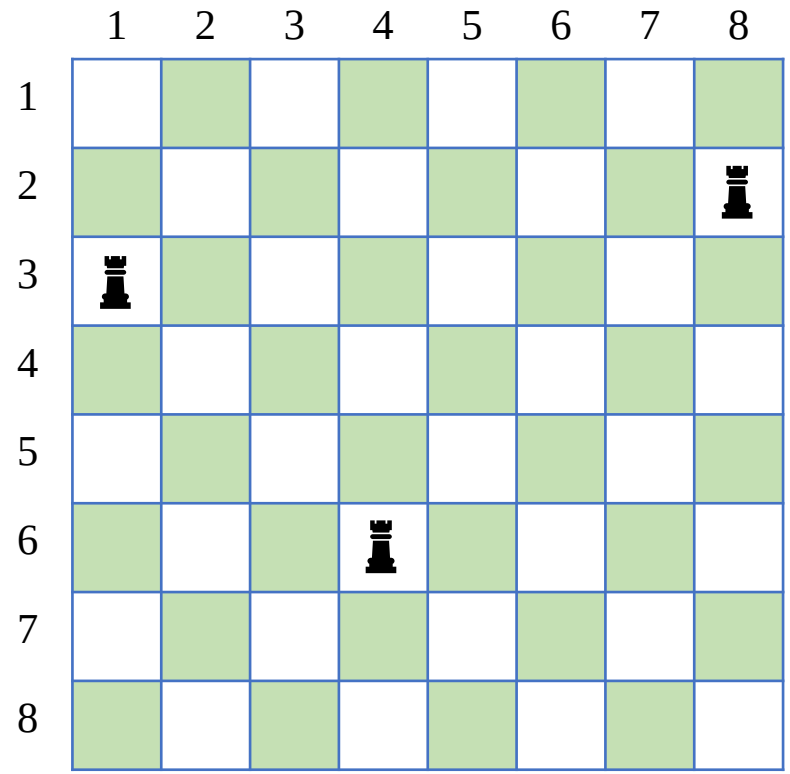
- *n*-Queens 문제의 해결
  - 기본 가정: 같은 행(row)에는 퀸을 놓지 않는다.
  - 유망 함수의 구현
    - 같은 열(column)이나 같은 대각선(diagonal)에 놓이는 지를 확인

5.2 n-Queens 문제의 구현

- 유망의 조건 1: 같은 열 체크
  - `col[i]`: i번째 행(row)에서 퀸이 놓여있는 열(column)의 위치
  - `col[k]`: k번째 행(row)에서 퀸이 놓여있는 열(column)의 위치
  - `col[i]==col[k]`: 같은 열에 놓이게 되므로, 유망하지 않다.

5.2 n-Queens 문제의 구현

- 유망의 조건 2: 대각선 체크




```
col[6] - col[3] = 4 - 1 = 3 = 6 - 3
col[6] - col[2] = 4 - 8 = -4 = 2 - 6
```

5.2 n-Queens 문제의 구현

- 유망의 조건 2: 대각선 체크
  - 왼쪽에서 위협하는 퀸에 대해서
    - 열에서의 차이는 행에서의 차이와 같다.
    - `col[i] - col[k] == i - k`
  - 오른쪽에서 위협하는 퀸에 대해서
    - 열에서의 차이는 행에서의 차이의 마이너스값과 같다.
    - `col[i] - col[k] == k - i`

  - `col[i]`와 `col[k]`의 절대값으로 대각선 위협 판단

5.2 n-Queens 문제의 구현

Algorithm 5.1: Backtracking Algorithm for the n-Queens Problem

```
def n_queens (i, col):
    n = len(col) - 1
    if (promising(col, i)):
        if (i == n):
            print(col[1: n + 1])
        else:
            for j in range(1, n + 1):
                col[i + 1] = j
                n_queens(col, i + 1)
```

5.2 n-Queens 문제의 구현

Algorithm 5.1: Backtracking Algorithm for the n-Queens Problem

```
def promising (i, col):
    k = 1
    flag = True
    while (k < i and flag):
        if (col[i] == col[k] or abs(col[i] - col[k]) == (i - k)):
            flag = False
        k += 1
    return flag
```

5.2 n-Queens 문제의 구현


```
n = 4
col = [0] * (n + 1)
n_queens(0, col)
```

```
[2, 4, 1, 3]
[3, 1, 4, 2]
```

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