TMItalk: Too Much Information Talk

8-퍼즐 문제와 A* 알고리즘

3. A* 알고리즘으로 풀어 보자.

TMItalk: 8-퍼즐 문제와 A* 알고리즘

- 8-퍼즐 문제를 A* 알고리즘으로 풀기:
  - A* 알고리즘은 다익스트라 알고리즘과 거의 유사하지만,
    - 휴리스틱 함수를 통해 [최단 거리 + 추정 거리]로 경로 선택
    - 우선순위: $f(x) = g(x) + h(x)$

1	2	3
4	5	6
7	8	0

출발 정점

$g(x)$ .....

3	6	8
0	5	7
2	1	4

경유 정점

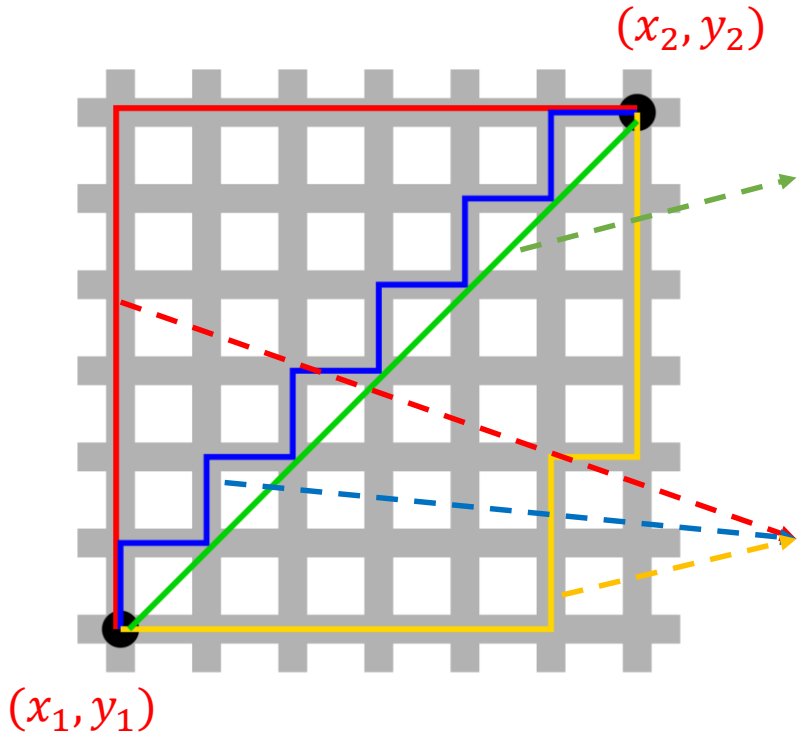
$h(x)$ .....

0	8	7
6	5	4
3	2	1

도착 정점

```
def astar_search(start, target):
    length, bypass = {start: 0}, {start: start}
    heap = []
    heapq.heappush(heap, (h(start, target), start))
    while len(heap) > 0:
        u = heapq.heappop(heap)[1]
        if u == target:
            return bypass
        for v in get_neighbors(u):
            if g(v, length) < 0:
                continue # skip vertices already visited.
            elif g(v, length) > g(u, length) + 1:
                length[v] = length[u] + 1
                bypass[v] = u
                heapq.heappush(heap, (length[v] + h(v, target), v))
        length[u] = -1 # mark u as visited
    return bypass
```

- **추정 거리를 위한 휴리스틱:**
  - 맨하탄 거리: Manhattan Distance
    - 유클리드 좌표계에서 두 점 사이의 각 차원의 절댓값의 합



$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$d = |(x_2 - x_1)| + |(y_2 - y_1)|$$

- 8-퍼즐에의 맨하탄 거리:
  - 한 타일의 현재 위치과 목표 위치간의 이동 거리

(row_1, col_1)

1	2	3
4	5	6
7	8	0

현재 위치

목표 위치

0	8	7
6	5	4
3	2	1

(row_2, col_2)

$$d = |(row_2 - row_1)| + |(col_2 - col_1)|$$

TMItalk: 8-퍼즐 문제와 A* 알고리즘

- 8-퍼즐을 위한 휴리스틱 함수:
  - $h(x)$: 모든 타일의 맨하탄 거리의 합

```
def h(start, target):
    # return heuristic distance using the Manhattan distance
    s, t = int_to_state(start), int_to_state(target)
    distance = 0
    for i in range(1, N):
        spos, tpos = s.index(i), t.index(i)
        srow, scol = spos // n, spos % n
        trow, tcol = tpos // n, tpos % n
        distance += abs(srow - trow) + abs(scol - tcol)
    return distance
```

- 공간 복잡도를 줄이기 위한 꼼수:
  - A*는 휴리스틱하게 유망한 상태들만 탐색하므로
    - 모든 정점의 최단 거리와 경유 상태를 저장할 필요가 없음
  - 딕셔너리(또는 해시맵)을 이용해서 최단 거리와 경유 상태 저장

```
length, bypass = {start: 0}, {start: start}

length[v] = length[u] + 1
bypass[v] = u


def g(v, length):
    return length[v] if v in length else INF
```

TMItalk: 8-퍼즐 문제와 A* 알고리즘

- 15-퍼즐로 확장할 때의 상태 전환 문제:
  - 15-퍼즐일 경우, 상태에서 정수로, 정수에서 상태로 전환하는데 문제 발생
    - [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 0]
    - 1234567891011121314 0
  - 해결하는 방법:
    - 정수로 바꿀 때 모두 두 자리수로 바꿔서 문자열이나 큰 정수로 변환

```python
def state_to_int(state):
    s = ""
    if N < 10:
        for i in range(len(state)):
            s += str(state[i])
        return int(s)
    else:
        for i in range(len(state)):
            s += str(state[i]).zfill(2)
        return int(s)
```

```python
def int_to_state(v):
    s = str(v)
    if N < 10:
        if len(s) != N:
            s = "0" + s
        return list(map(int, s))
    else:
        while len(s) < 2 * N:
            s = "0" + s
        state = []
        for i in range(0, len(s), 2):
            state.append(int(s[i:i+2]))
        return state
```

TMItalk: 8-퍼즐 문제와 A* 알고리즘

- A* 알고리즘은 15-퍼즐을 풀 때 얼마나 걸릴까?

```
puzzle.8.4.in:


16
00 15 14 13
12 11 10 09
08 07 06 05
04 03 02 01
```

```
00 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01
15 00 14 13 12 11 10 09 08 07 06 05 04 03 02 01
15 11 14 13 12 00 10 09 08 07 06 05 04 03 02 01
15 11 14 13 00 12 10 09 08 07 06 05 04 03 02 01
15 11 14 13 08 12 10 09 00 07 06 05 04 03 02 01

……

01 02 03 04 05 06 07 08 09 00 10 11 13 14 15 12
01 02 03 04 05 06 07 08 09 10 00 11 13 14 15 12
01 02 03 04 05 06 07 08 09 10 11 00 13 14 15 12
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 00
the number of steps to solve it: 346
```

Any Questions?

주니온TV@Youtube

자세히 보면 유익한 코딩 채널