03강

인공지능

문제풀이(2)

컴퓨터과학과 이병래 교수

학습목차

01

경험적 탐색의 개념

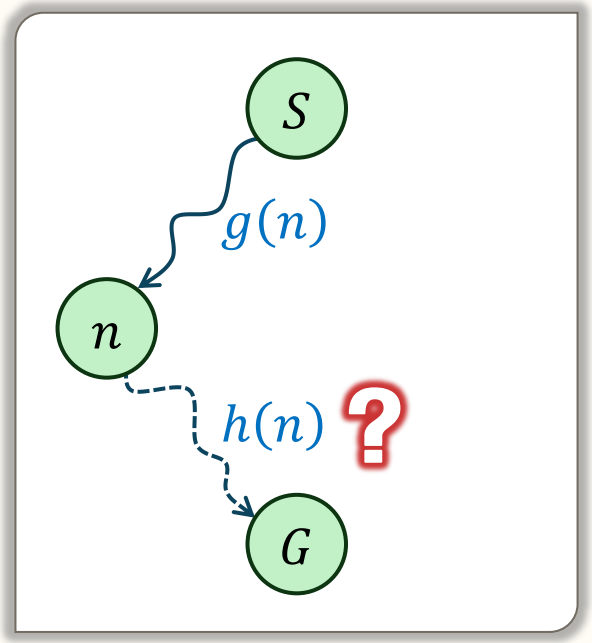
1. 경험적 탐색(heuristic search)이란?

- 목표상태를 보다 신속하게 탐색하기 위해 **경험적 규칙**을 사용하는 탐색 방법
  - **경험적 규칙**(rule of thumb): 항상 옳은 것은 아니지만 대부분의 경우에 잘 맞는 규칙
- 경험적 규칙을 **평가함수**에 반영
  - **평가함수**(evaluation function): 어떤 상태가 목표상태의 탐색에 바람직한 정도를 평가하기 위한 척도
    - 해를 향해 가는데 필요한 비용, 해로 향하는 경로상에 존재할 가능성 등

2. 평가함수

❑ 평가함수의 구성 요소

- 출발노드 S에서 출발하여 노드 n까지 도착하였을 때, 노드 n의 평가함수의 정의에 포함될 수 있는 비용



- $g(n)$: 출발노드 S로부터 현재상태를 나타내는 노드 n까지 도달하는 데 소비한 경로비용
- $h(n)$: 노드 n으로부터 목표노드 G까지 도달하는 데 필요한 경로비용

⇨ $\hat{h}(n)$: 경험적 지식을 이용하여 $h(n)$을 예측한 비용

02

언덕오르기 탐색

1. 언덕오르기 탐색 알고리즘

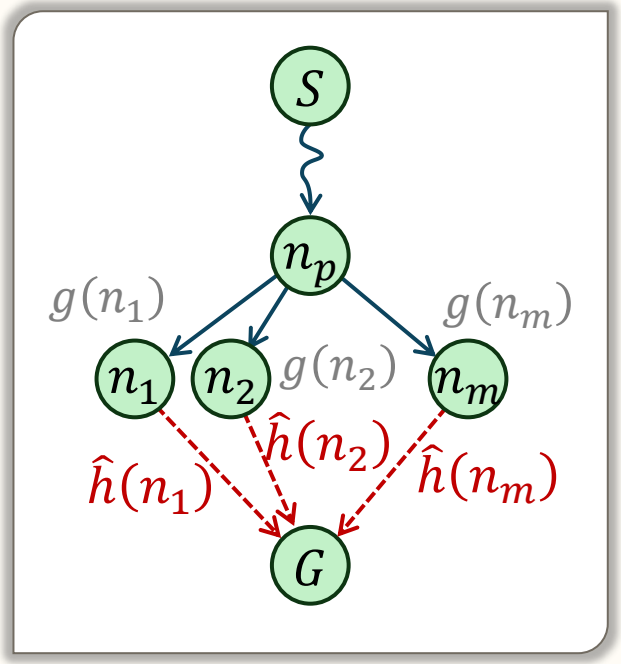
탐색 순서

- 현재상태를 확장하여 생성된 후계노드들 중에서 다음 확장할 노드를 선택함(깊이우선 탐색과 유사한 순서로 탐색)

선택 기준: 평가함수로 계산한 비용이 최소인 노드를 선택

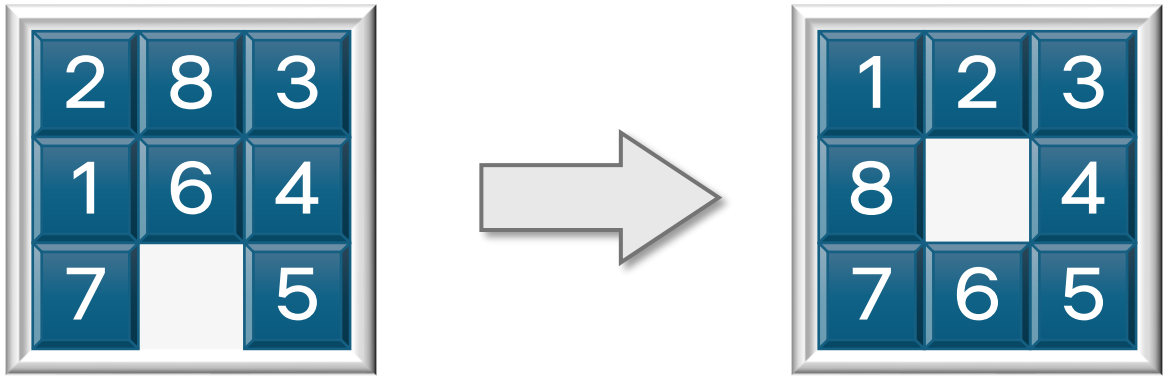
평가함수 $= \hat{h}(n)$

- 후계노드 n으로부터 목표노드 G에 도달하는 비용을 예측한 값
- 후계노드까지 도달하는데 사용된 비용 $g(n)$은 고려하지 않음



2. 예제: 8-퍼즐 문제의 풀이

8-퍼즐 문제



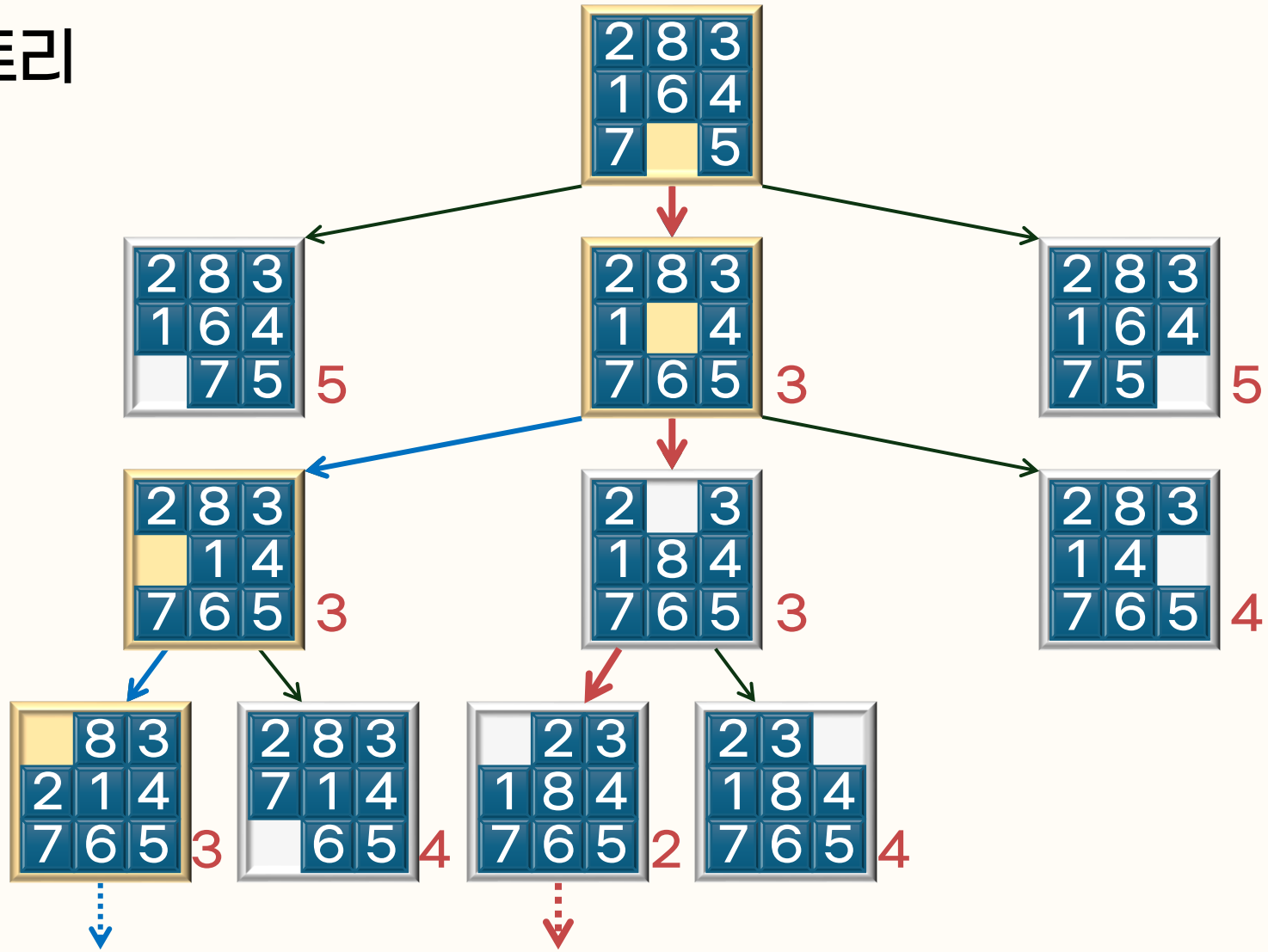
초기상태 목표상태

- 어떠한 상태의 비용: 목표상태의 퍼즐과 비교했을 때 지정된 위치에 존재하지 않는 조각의 수

  ⇨ 초기상태의 비용 = 4

2. 예제: 8-퍼즐 문제의 풀이

- 탐색 트리

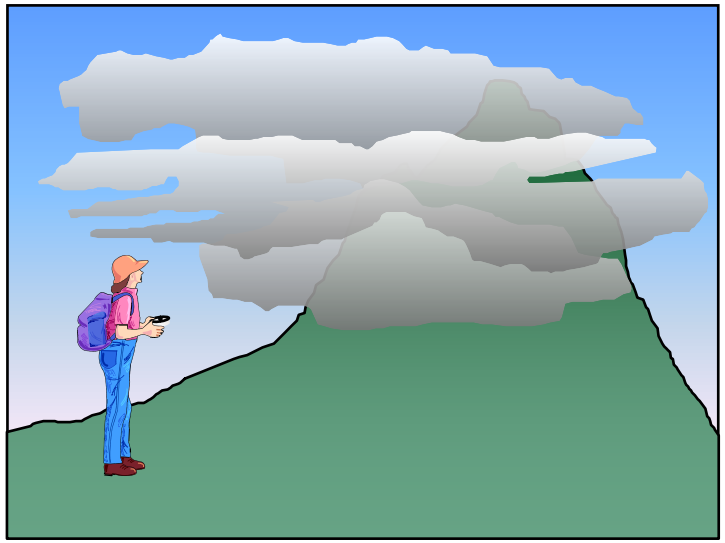


3. 계수최적화 문제

예제

등산 문제

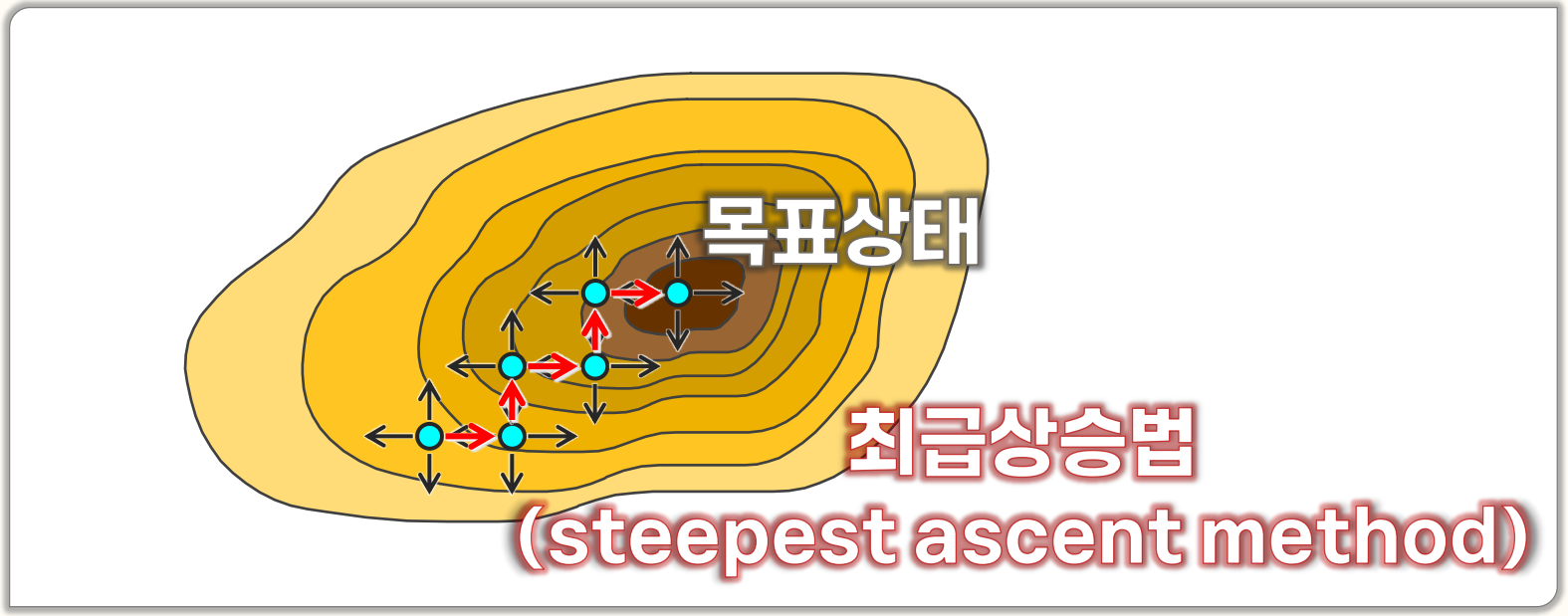
어떤 사람이 초행길의 산을 등산하는 도중 짙은 안개를 만났다. 지도도 없고, 사람이 다닌 길도 없으며, 오직 나침반에만 의지하여 산에 올라간다. 정상에 도달하려면 어떻게 해야 하는가?



3. 계수최적화 문제

❑ 등산 문제의 풀이

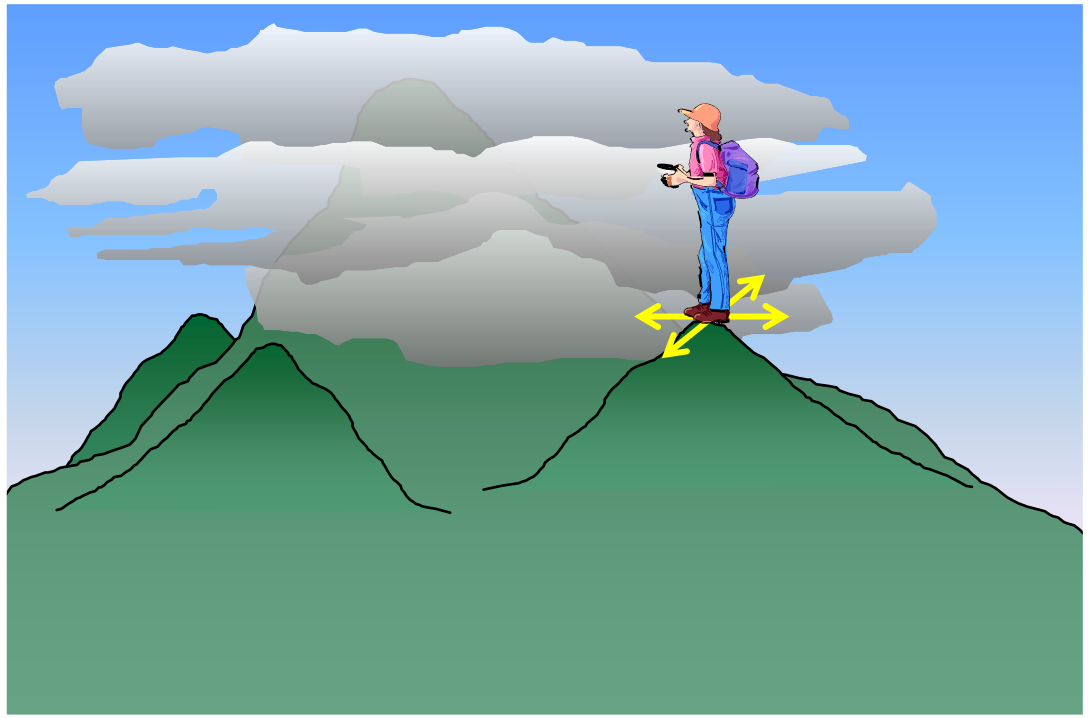
- 상태: 등산가의 좌표 및 고도
- 연산자: 동, 서, 남, 북 방향으로 정해진 거리만큼 이동
- 목표상태: 모든 후계상태의 고도가 현재상태보다 낮은 상태



3. 계수최적화 문제

❑ 최급상승법의 난제

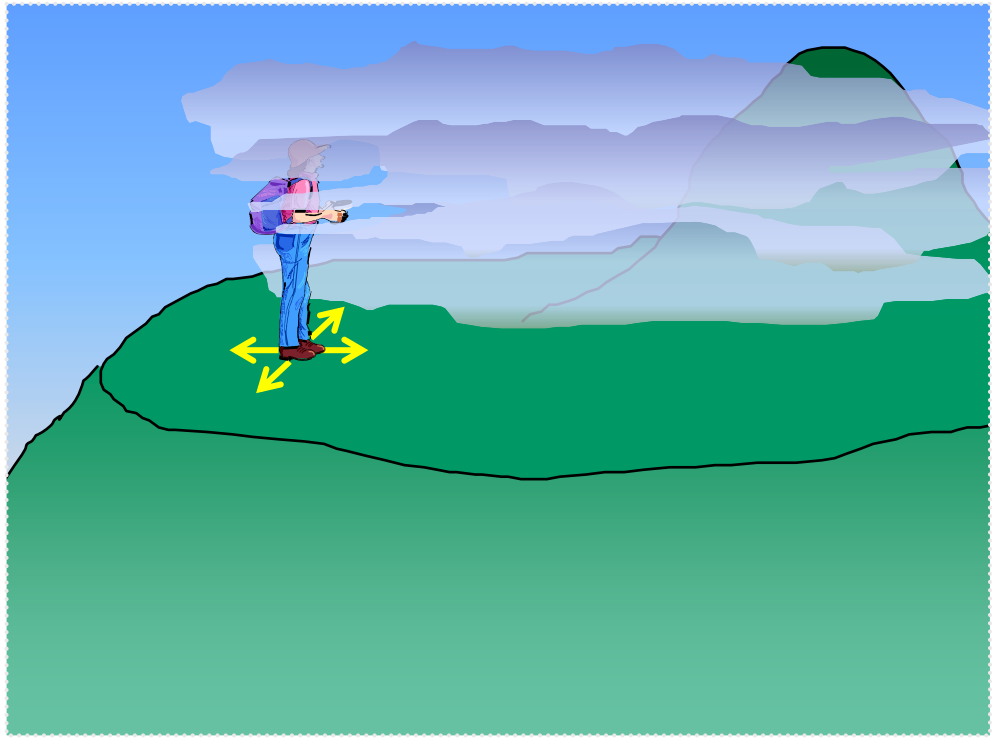
지역최대치 문제 | 고원문제 | 능선문제



3. 계수최적화 문제

❑ 최급상승법의 난제

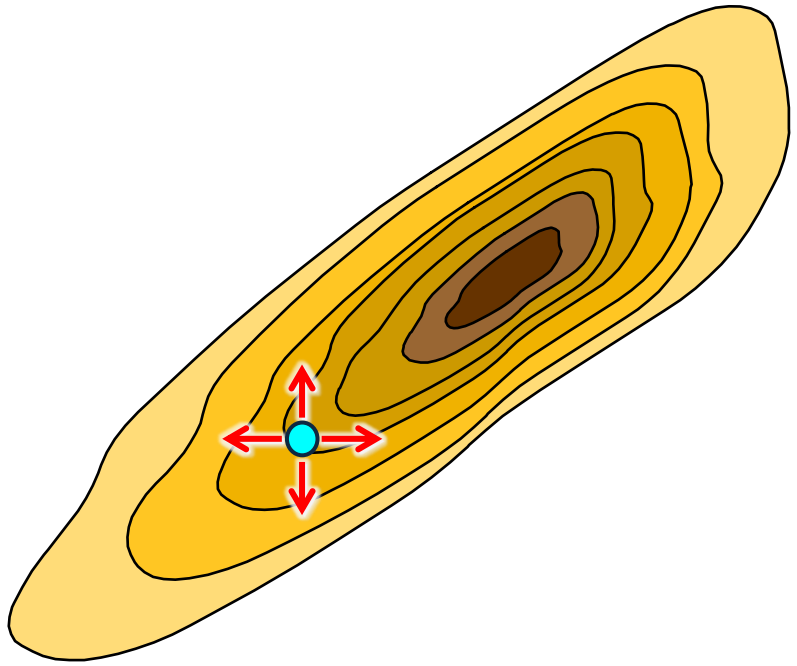
지역최대치 문제 | 고원문제 | 능선문제



3. 계수최적화 문제

❑ 최급상승법의 난제

지역최대치 문제 | 고원문제 | 능선문제



03

모의 담금질

1. 모의 담금질의 개념

❑ 모의 담금질(simulated annealing)이란?

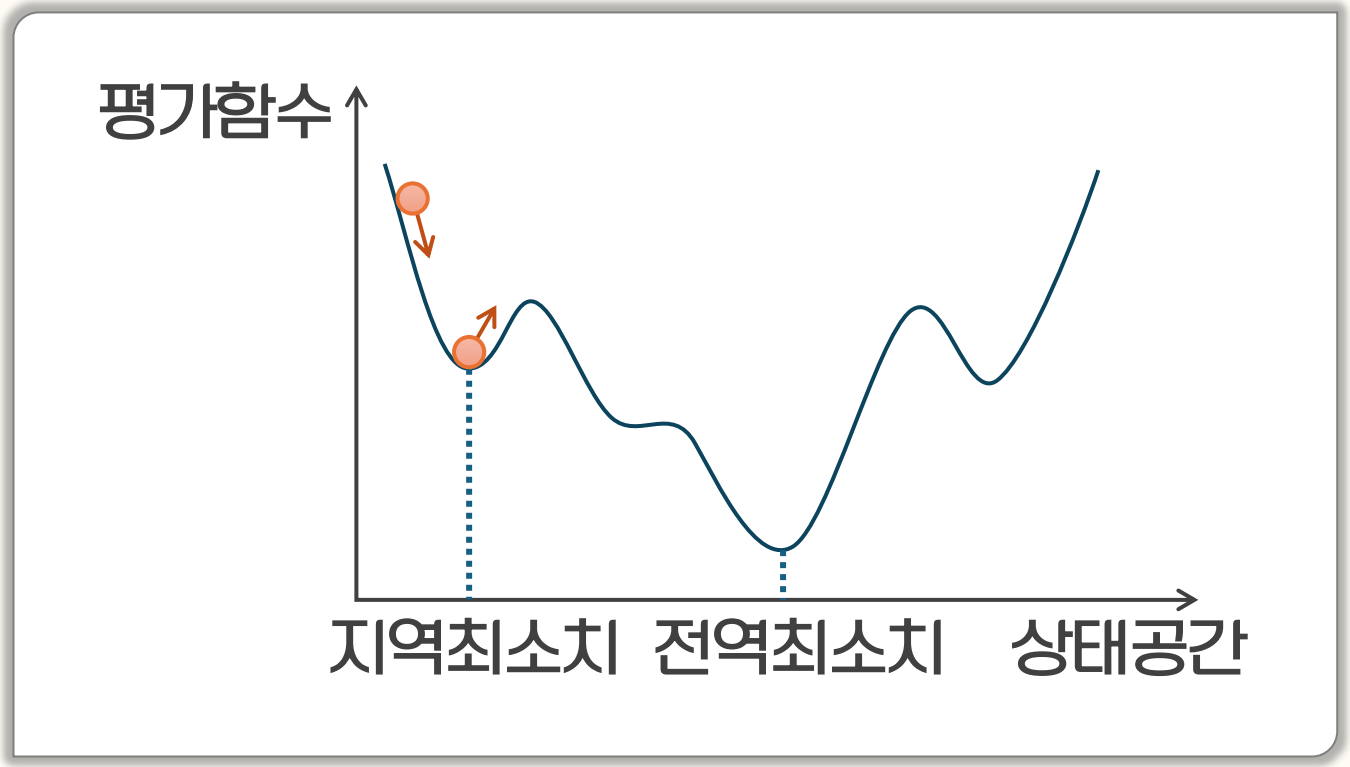
- 평가함수의 값이 전역최소치(또는 전역최대치)에 해당되는 해를 구하기 위한 확률적 접근방법

annealing(풀림) : 금속이나 유리를 일정한 온도로 가열한 다음에 천천히 식혀 내부 조직을 고르게 하고 응력(應力)을 제거하는 열처리 조작

1. 모의 담금질의 개념

❑ 모의 담금질(simulated annealing)이란?

- 평가함수의 값이 전역최소치(또는 전역최대치)에 해당되는 해를 구하기 위한 확률적 접근방법



2. 모의 담금질 알고리즘

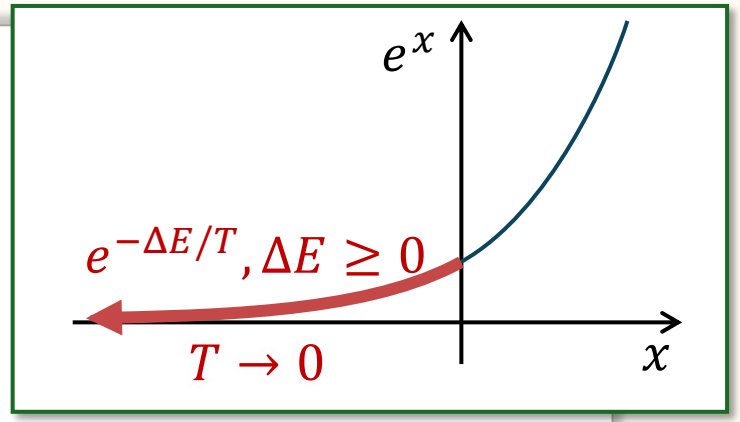
모의 담금질 알고리즘

```
1  // temperature(t): 시간 t에 따른 온도를 나타내며,
2  //                 t에 따라 서서히 감소하도록 함
3  // h(s): 상태s에 대한 평가함수
4  현재상태 ← 문제의 초기상태;
5  for t=1 to ∞ do
6      T = temperature(t);
7      if T == 0 then
8          return 현재상태;
9      end-if;
10     차기상태 = 현재상태의 후계노드 중에서 임의로 선택;
11     ΔE = h(차기상태) - h(현재상태);
12     if ΔE < 0 then
13         ......
```

2. 모의 담금질 알고리즘

모의 담금질 알고리즘

```
5  for t=1 to ∞ do
6      T = temperature(t);
7      if T == 0 then
8          return 현재상태;
9      end-if;
10     차기상태 = 현재상태의 후계노드 중에서 임의로 선택;
11     ΔE = h(차기상태) - h(현재상태);
12     if ΔE < 0 then
13         현재상태 = 차기상태;
14     else
15         확률 e^-ΔE/T에 따라 차기상태를 현재상태로 선택;
16     end-if;
17 end-for;
```



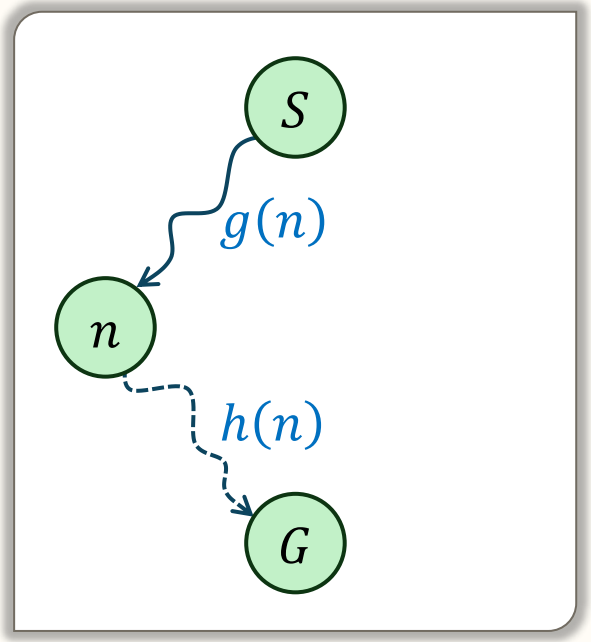
04

A* 알고리즘

1. A* 알고리즘의 평가함수

❑ 평가함수의 구성 요소

- 노드 n까지 도달한 상태에서 출발노드 S에서 노드 n을 거쳐 목표노드 G까지 도달하는 전체 경로의 비용 계산



- $g(n)$: 출발노드 S로부터 현재상태를 나타내는 노드 n까지 도달하는 데 소비한 경로비용
- $h(n)$: 노드 n으로부터 목표노드 G까지 도달하는 데 필요한 경로비용

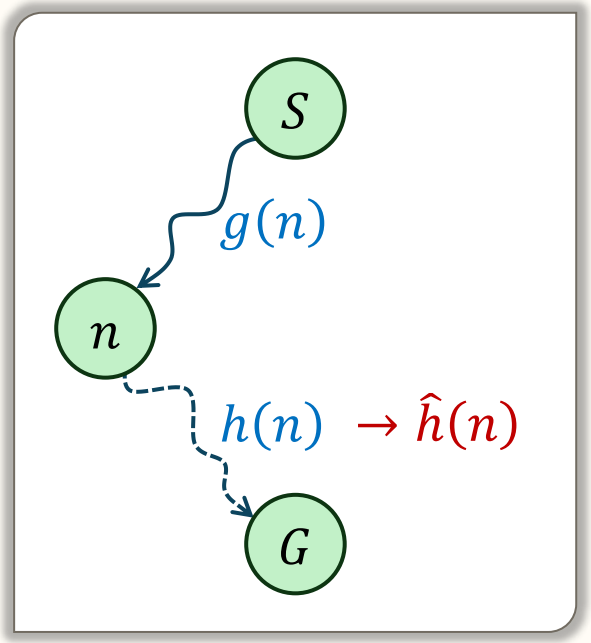
➡ 전체 경로비용 $f(n)$

$$f(n) = g(n) + h(n)$$

1. A* 알고리즘의 평가함수

❑ 평가함수의 구성 요소

- 노드 n까지 도달한 상태에서 출발노드 S에서 노드 n을 거쳐 목표노드 G까지 도달하는 전체 경로의 비용 계산



- $g(n)$: 출발노드 S로부터 현재상태를 나타내는 노드 n까지 도달하는 데 소비한 경로비용
- $\hat{h}(n)$: 노드 n으로부터 목표노드 G까지 도달하는 데 드느 비용의 예측치

➡ 평가함수 $\hat{f}(n)$: $f(n)$의 예측치

$$\hat{f}(n) = g(n) + \hat{h}(n)$$

2. A* 알고리즘

A* 알고리즘

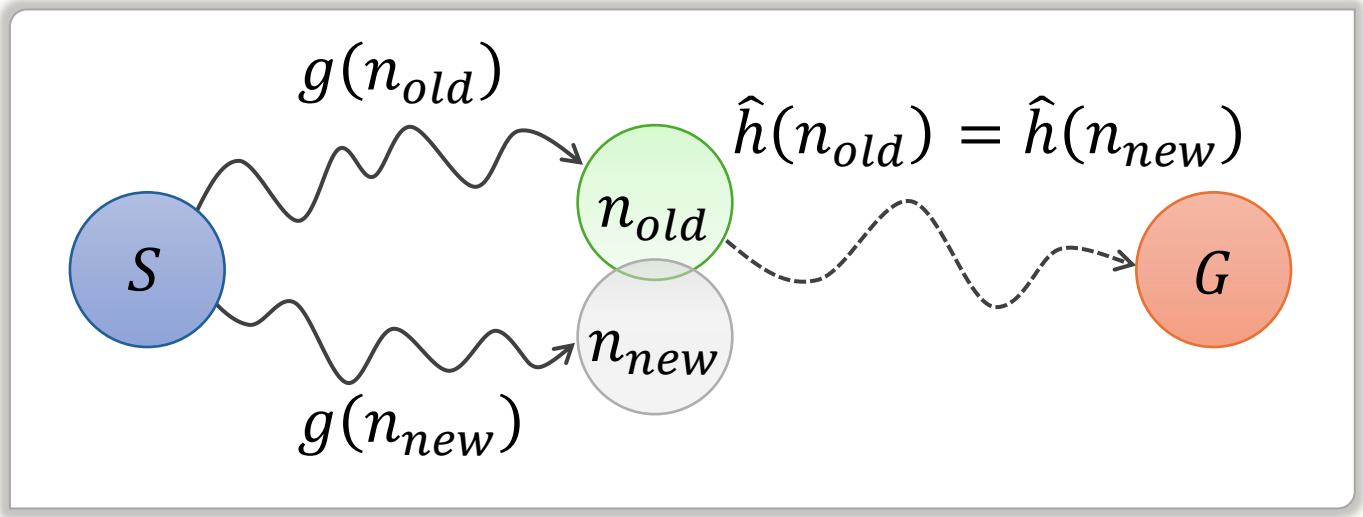
1. 출발노드 S를 OPEN에 삽입($\hat{f}(S)$를 계산하여 첨부)
2. OPEN에 남은 노드가 있으면 다음을 반복
   1) OPEN에서 $\hat{f}$이 최소인 노드 n을 꺼내 CLOSED에 넣는다.
   2) 노드 n이 목표노드라면 탐색성공
   3) 노드 n을 확장하여 후계노드 $n_1, n_2, \cdots, n_m$을 생성
   4) 후계노드의 평가함수 $\hat{f}(n_1), \hat{f}(n_2), \cdots, \hat{f}(n_m)$을 계산
   5) 후계노드 $n_i, i = 1, 2, \cdots, m$을 OPEN에 삽입
      (중복 생성된 노드 제거)
3. 탐색 실패

2. A* 알고리즘

중복 생성된 노드 n_{new}의 처리

- 동일한 상태(노드 n_{old})가 OPEN에 존재하는 경우

➡ 아직 어느 노드도 확장되지 않은 상태이므로 평가함수 $\hat{f}$이 큰 노드를 제거하면 됨



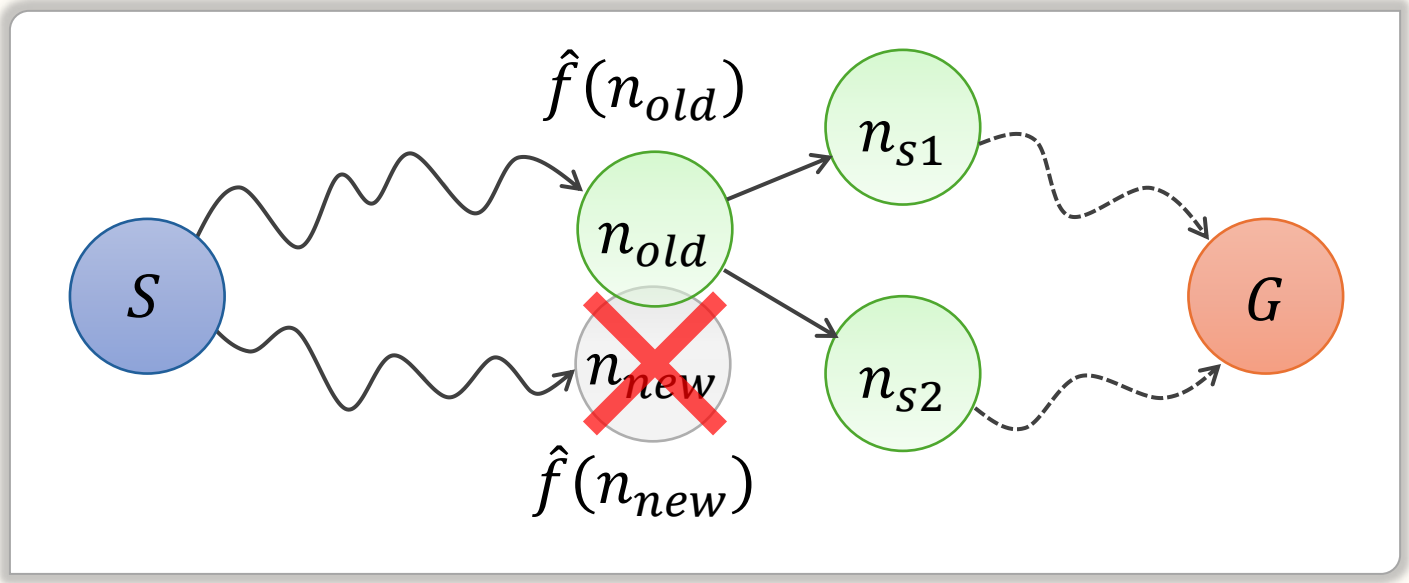
2. A* 알고리즘

❑ 중복 생성된 노드 n_{new}의 처리

- 동일한 상태(노드 n_{old})가 CLOSED에 존재하는 경우

① $\hat{f}(n_{old}) \leq \hat{f}(n_{new})$인 경우

⇨ n_{new}를 제거하면 됨



2. A* 알고리즘

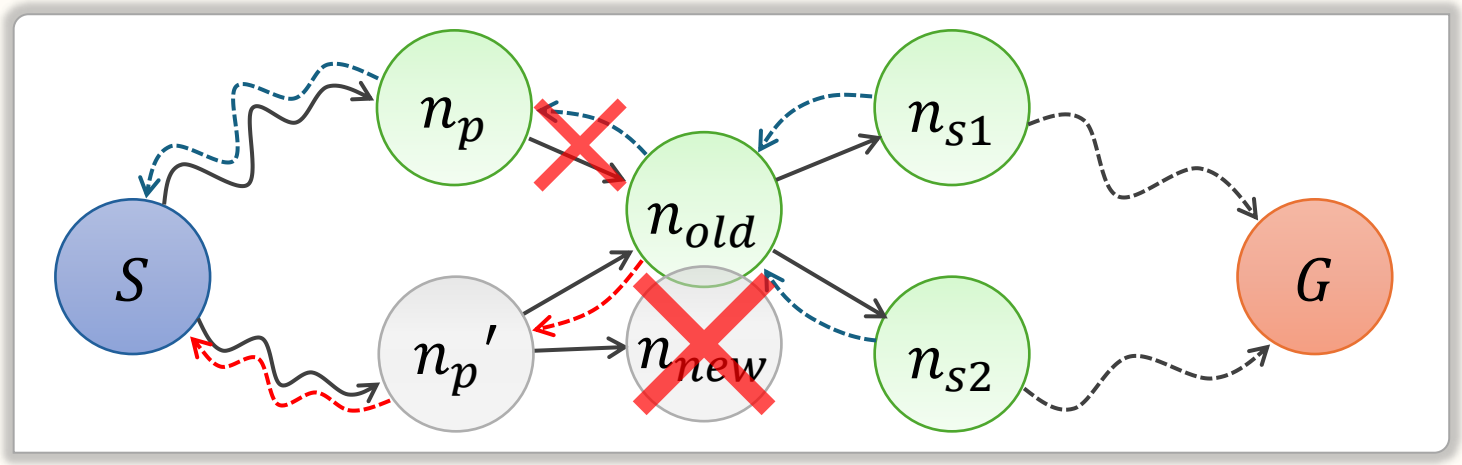
❑ 중복 생성된 노드 n_{new}의 처리

- 동일한 상태(노드 n_{old})가 CLOSED에 존재하는 경우

② $\hat{f}(n_{old}) > \hat{f}(n_{new})$인 경우

➡ n_{new}를 제거하되 n_{old}의 부모노드 포인터가 n_{new}의 부모노드를 가리키도록 수정

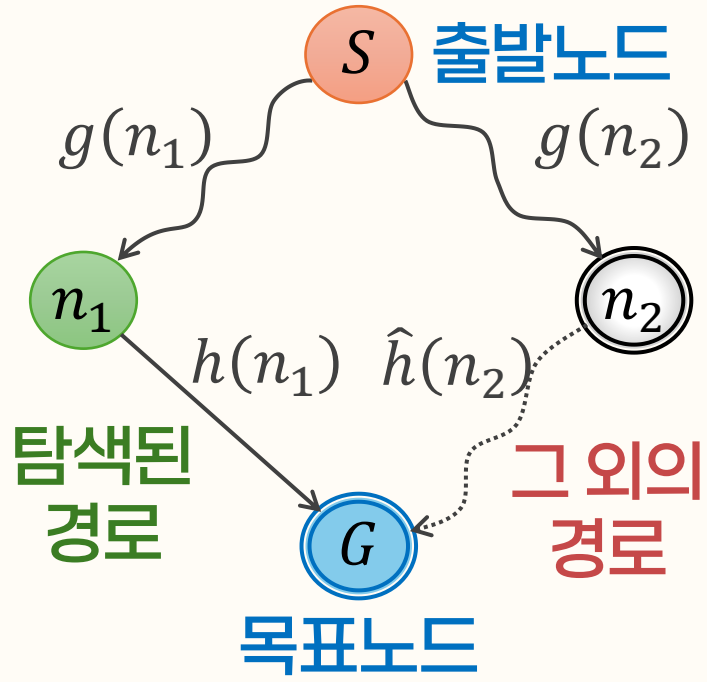
➡ n_{old} 및 n_{old}의 후계노드들의 평가함수 값을 갱신



2. A* 알고리즘

❑ 탐색된 경로가 최소비용경로가 되기 위한 조건

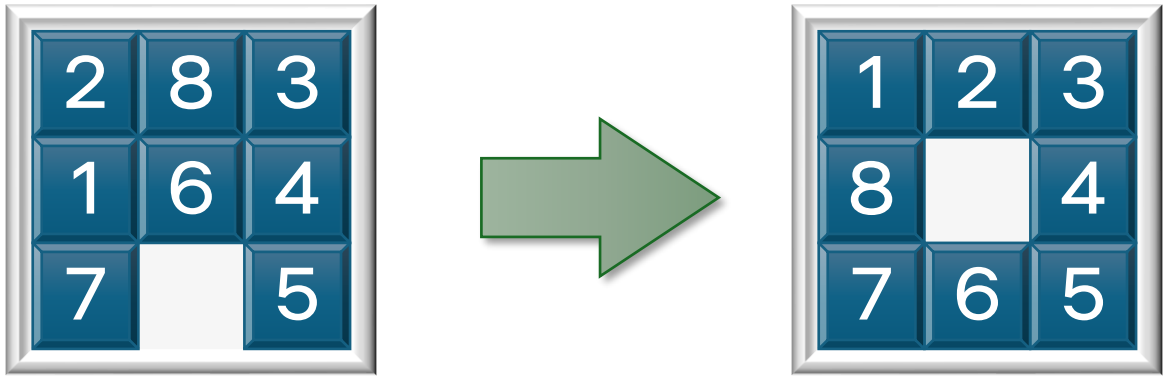
만일 어느 경우에도 $\hat{h}$을 h보다 큰 값으로 예측하지 않는다면 (즉, 항상 $\hat{h}(n) \leq h(n)$이 성립함), A* 알고리즘은 최소비용 경로를 탐색하는 것을 보장한다.



$$g(n_1) + h(n_1) \leq g(n_2) + \hat{h}(n_2)$$
$$\leq g(n_2) + h(n_2)$$

3. 예제: 8-퍼즐 문제의 풀이

8-퍼즐 문제

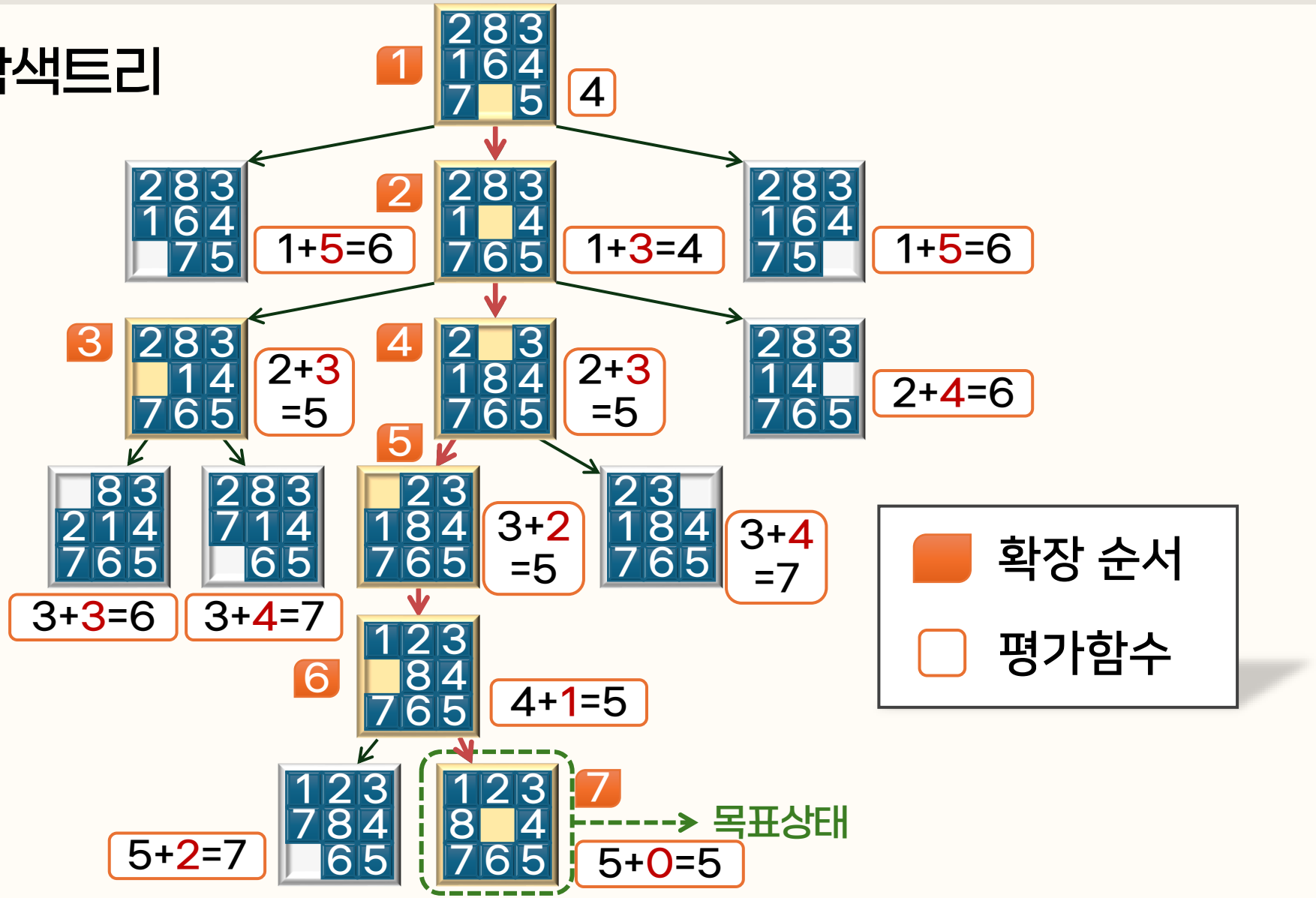


초기상태 목표상태

- $g(n)$: 빈 칸의 이동 횟수
- $\hat{h}(n)$: 목표상태의 퍼즐과 비교했을 때 지정된 위치에 존재하지 않는 조각의 수

3. 예제: 8-퍼즐 문제의 풀이

탐색트리



4. 예제: 경로 탐색 문제의 풀이

그림 1

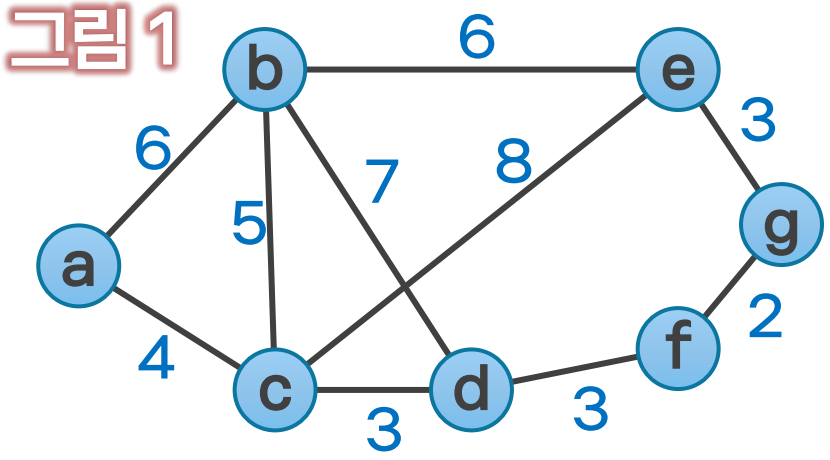
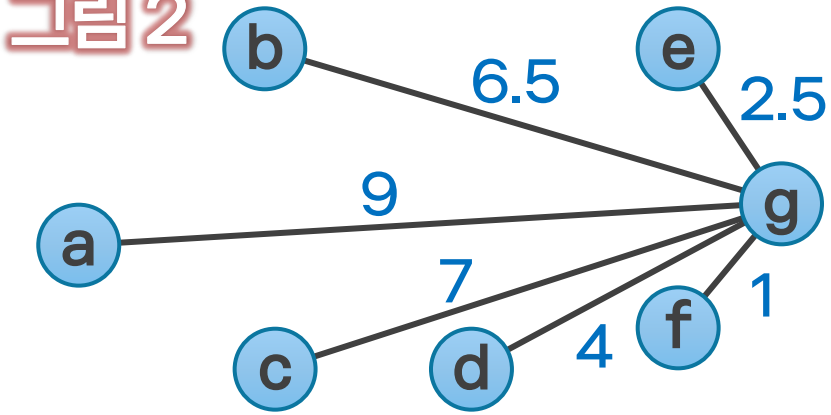
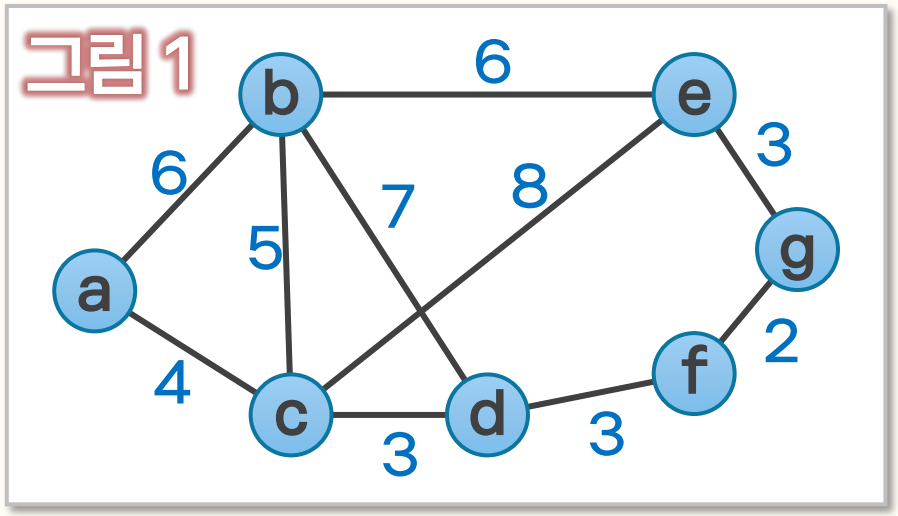


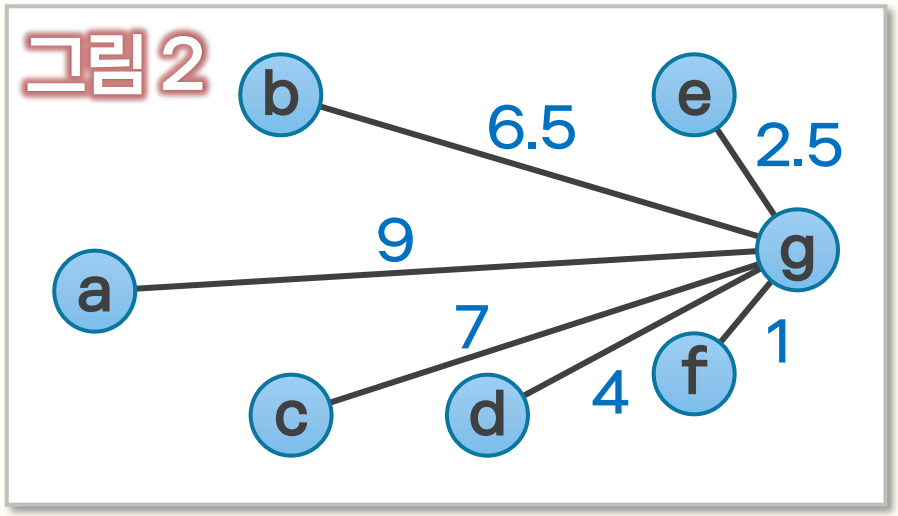
그림 2

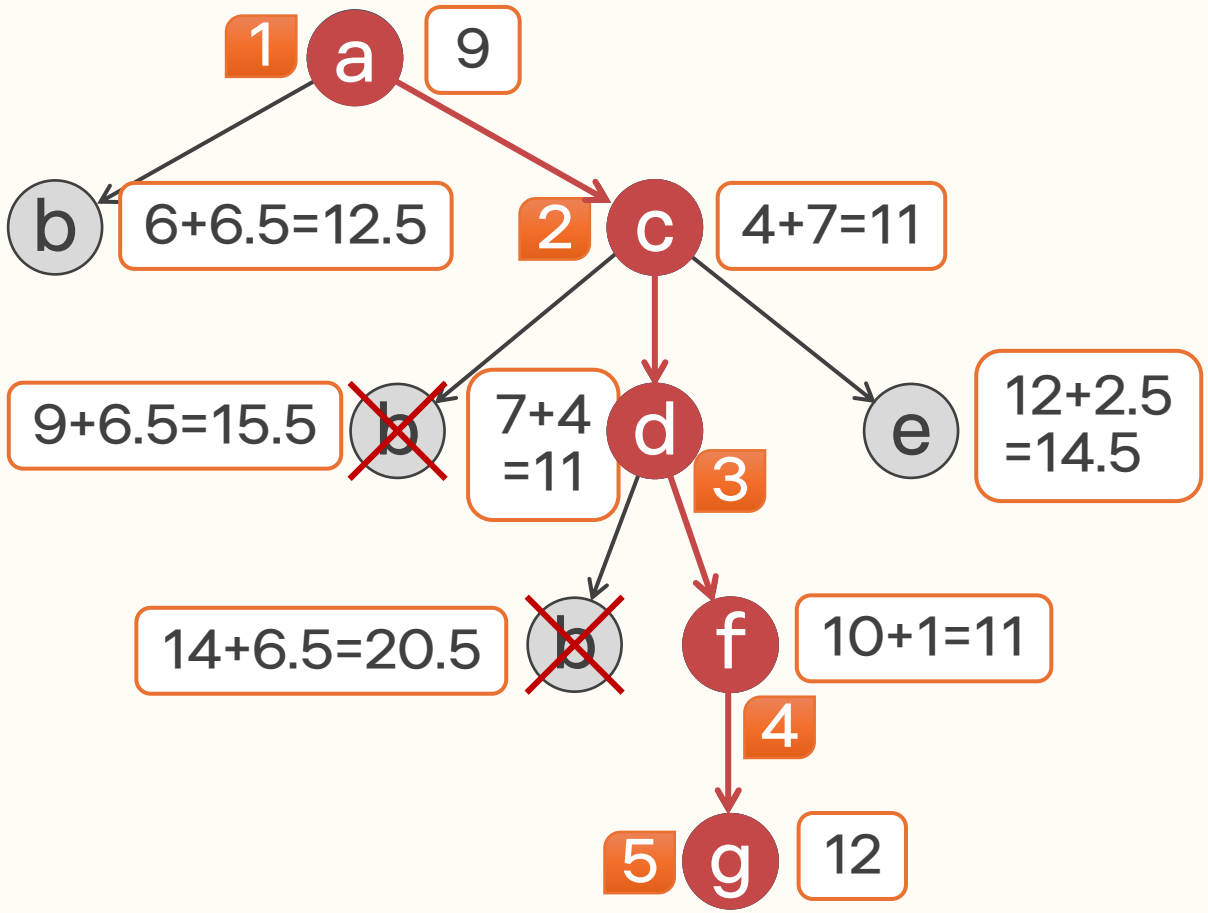


a, b, c, d, e, f, g라는 7개의 도시를 연결하는 도로망이 건설되어 있다. 어떤 여행자가 도시a를 출발하여 도시g까지 가는 경로를 찾고자 한다. [그림1]은 각 도시를 연결하는 도로와 그 거리를 표현한 그래프이고, [그림2]는 각 도시로부터 목적지인 g까지의 직선거리이다. A* 알고리즘을 이용하여 최단길이 경로를 탐색하라.

4. 예제: 경로 탐색 문제의 풀이







정리하기

- 경험적 탐색은 목표상태를 보다 효과적으로 탐색하기 위해 경험적 지식을 평가함수에 반영한다.
- 언덕오르기 탐색은 현재상태를 확장하여 생성된 후계노드 중에서 평가함수로 계산한 비용이 최소인 노드를 다음 확장할 노드로 선택한다. 이때 노드의 비용은 그 노드로부터 목표노드에 도달하는 비용을 예측한 값이다.
- 언덕오르기 탐색과 같은 계수 최적화 방법에서는 지역최대치 문제, 고원문제, 능선문제 등으로 인해 최적의 해에 해당되는 목표상태에 도달하지 못할 가능성이 있다.

정리하기

- 모의담금질(simulated annealing)은 시간에 따라 감소하는 확률에 따라 평가함수의 값이 개선되지 않는 후계상태로도 이동할 수 있게 하는 확률적 접근방법이다.
- A* 알고리즘에서 노드의 평가함수는 출발노드로부터 그 노드에 도달하기 위한 비용과 그 노드로부터 목표노드에 도달하는데 필요한 예측비용의 합으로 정의된다.
- A* 알고리즘에서는 평가함수가 최소인 노드를 선택하여 확장한다. 예측비용이 항상 실제 비용 이하로 예측되도록 정의한다면 탐색된 경로는 최소비용 경로이다.

04강

다음시간안내 ▶▶▶

게임트리