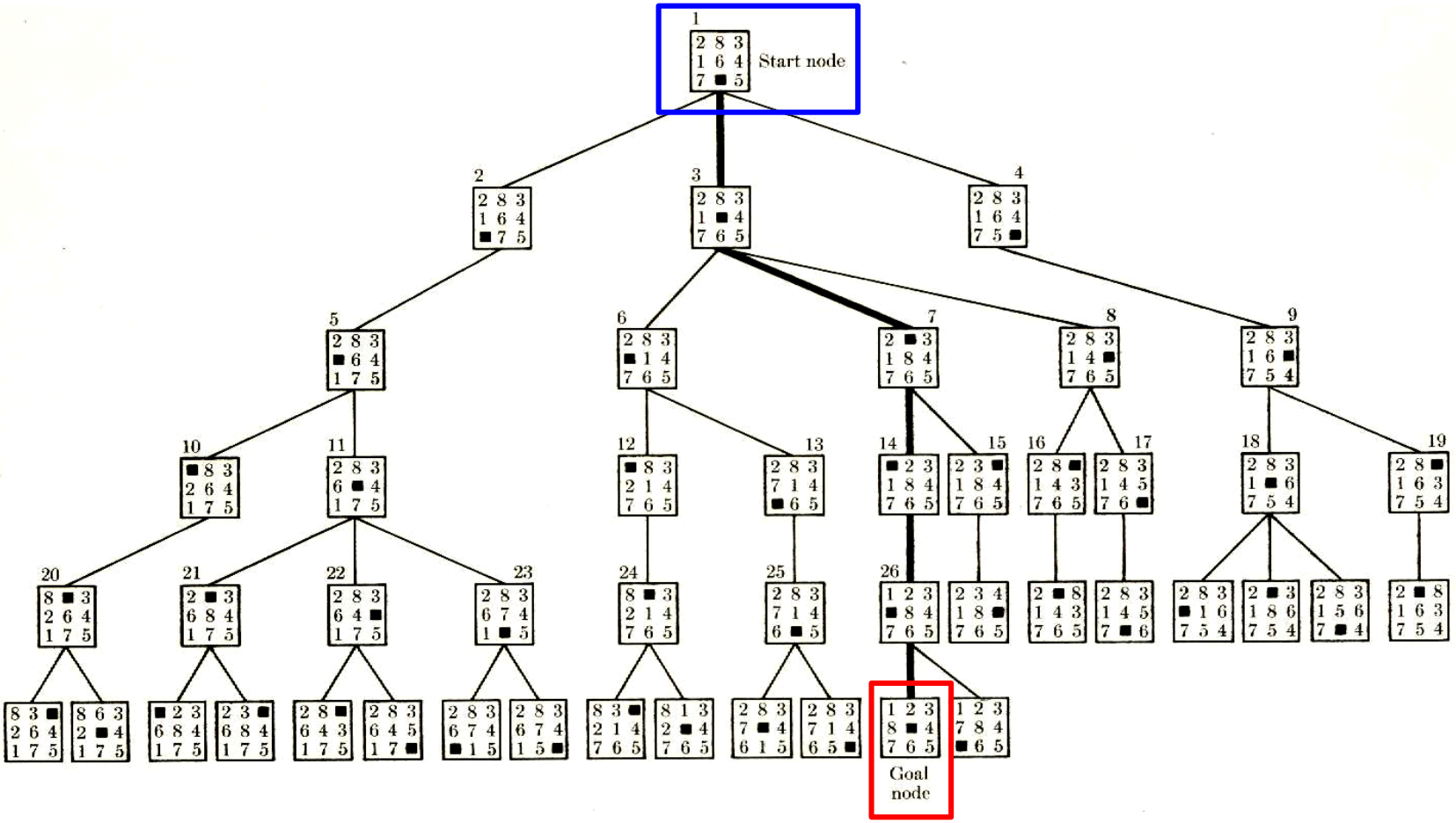
3장 탐색법 : 가장 효율적인 경로를 어떻게 선택할까?



출처: http://www.cse.buffalo.edu/~rapaport/572/S02/8-puzzle.html

3장 탐색법 : 가장 효율적인 경로를 어떻게 선택할까?

- **학습 목표**
  - 탐색법의 분류
  - 체계적 탐색
  - 휴리스틱 탐색
  - 탐색법 정리
  - 체험해 봅시다: Ex7_탐색법.xlsm

    (https://github.com/jpub/LearnAIwithExcel)

    최소 비용으로 산의 정상까지의 경로 탐색

3.1.1 탐색법의 개념

- **탐색법**은 문제 해결에 있어서 상태 공간 내에서의 초기 상태에서 목표 상태에 이르는 경로를 결정하는 방법을 의미.
- 앞 장의 문제 해결 개념에서는 연산자 적용에 동반되는 제약과 금지 상태를 피하는 것만을 생각하였지만, 탐색법에서는 다음과 같은 개념들도 함께 고려.

✓ **도달 보증:** 탐색이 순환하지 않고 어디선가 멈추는 것. 멈춘 곳이 해인지 아닌지는 관계없음.

✓ **최적해:** 목표 상태에 도달하는 것.

✓ **국소해:** 목표 상태 이외의 곳에 도달하는 것.

✓ **최적 경로:** 경로 비용을 고려할 때 목표 상태에 이르는 경로의 전체 비용이 최소가 되는 경로.

✓ **누적 비용:** 각 상태에 도달하기까지의 경로 비용(실제 값)

✓ **장래 비용:** 각 상태부터 목표 상태까지의 경로 비용(추측 값)

3.1.2 탐색법 전략에 따른 분류

- 전략의 방법에 따라 탐색법을 다음과 같이 분류.
  - **맹목적 탐색:** 적용 가능한 연산자를 무작위로 선택. 중복을 신경 쓰지 않음. 해가 있어도 반드시 도달한다고 보장할 수 없음.
  - **체계적 탐색:** 모든 상태 공간을(중복 없이) 조사. 해가 있으면 반드시 도달하나 비효율적임.
  - **휴리스틱 탐색:** 경험 법칙에 따른 상태 공간 축소를 수행하여 효율적인 탐색을 도모.
- 전략성이 전혀 없는 **맹목적 탐색**은 난수 등으로 진행 방향을 결정하게 되지만, 일상의 실제 문제에서는 이와 유사한 무계획적 행동이 많이 있음.

3.1.2 탐색법 전략에 따른 분류

- 하지만 적어도 중복은 피하거나 가급적 기대 효과가 큰 곳에서부터 손을 대고자 한다면, **어떤 평가 기준을 정하고 평가값이 좋은 쪽을 먼저 고려하는 것이 좋음**.
- 그 평가 기준으로서 다음과 같은 평가 함수를 생각.

식 6-1

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

e(n): 누적 비용 → 초기 상태부터 상태 n까지의(최소라고 생각하는) 실제 비용
h(n): 장래 비용 → 상태 n부터 목표 상태까지 예상되는 최소 비용(알 수 없는 경우에는 한 단계 앞의 비용을 사용)

3.1.2 탐색법 전략에 따른 분류

- 즉, **평가 함수**는 과거의 **실제 비용(누적 비용)**과 **장래 예측 비용(장래 비용)**의 합계.
- 그러나 이 모든 비용은 중간 상태에서는 진정한 값을 알 수 없음.
- 과거의 실제 비용 합계도 나중에 생각하면 더욱 좋은 경로가 있었음이 밝혀질 수도 있으므로 최적의 누적 비용일지 어떨지는 알 수 없음.
- **장래 비용**이라면 전적으로 **예측**할 수밖에 없음.
- 그럼에도 불구하고 평가 함수로서 식 6-1을 사용하려면 **누적 방식과 경험 법칙의 활용 등에 대한 연구가 필요.**

3.2 체계적 탐색

- 우선 전체 경로를 빠짐없이 모두 조사하는 것을 생각.
- 시간이 너무 많이 걸려 비현실적일지도 모르겠지만 하나씩 조사하면 가장 좋은 경로를 구할 수 있음.
- 이런 탐색법을 **체계적 탐색(Systematic Search)**이라고 부르며 다음과 같은 종류가 있음.
  - ✓ **깊이우선 탐색:** 스택(Stack)을 사용한 전체 경로 탐색. 비용은 생각하지 않으므로 평가 함수 없음.
  - ✓ **너비우선 탐색:** 큐(Queue)를 사용한 전체 경로 탐색. 비용은 생각하지 않으므로 평가 함수 없음.
  - ✓ **분기 한정법:** 전체 경로 탐색. 누적 비용이 적은 방향으로 진행. **최적 경로 보장**. 평가 함수: f(n) = e(n)

3.2 체계적 탐색

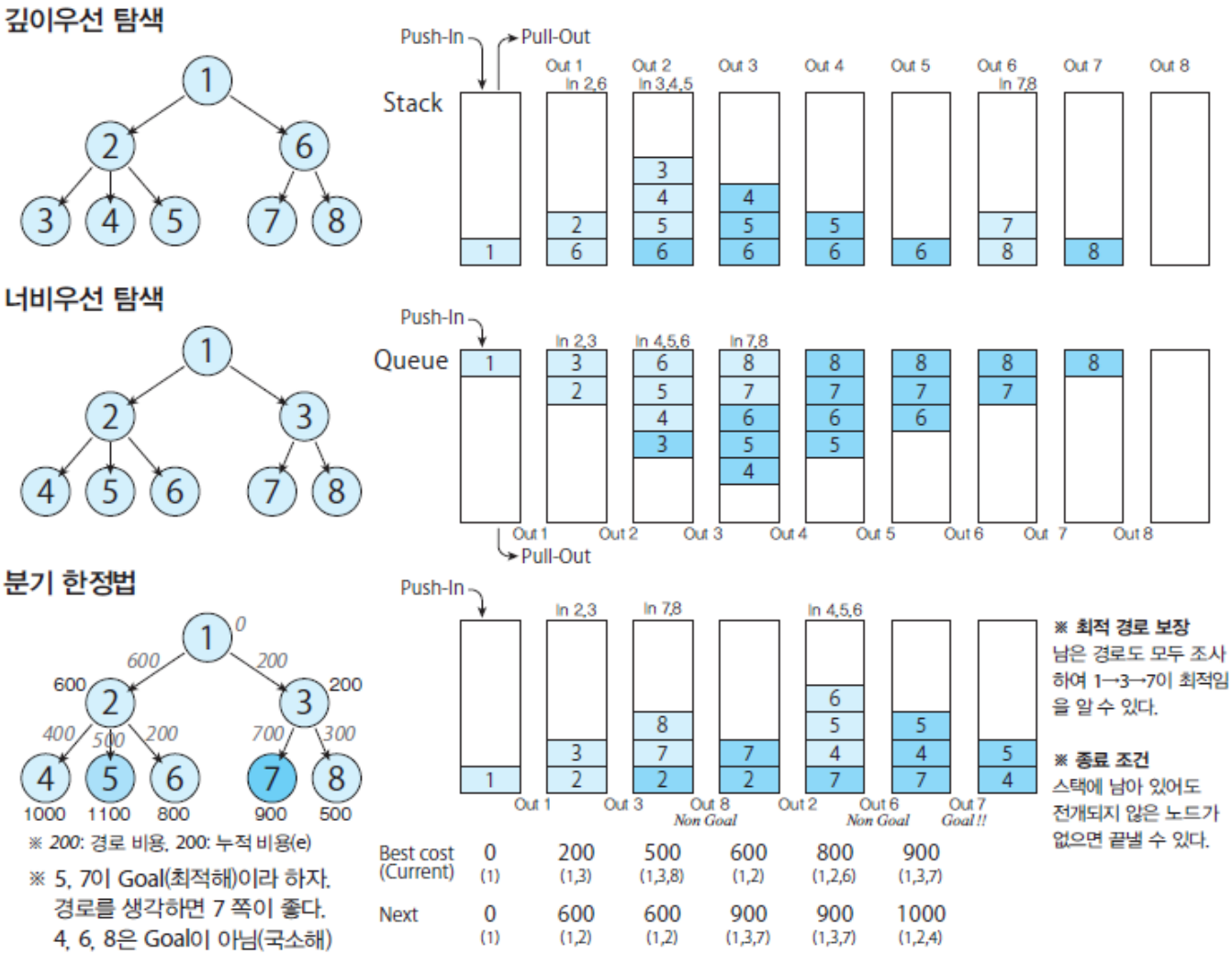


그림 6-2 체계적 탐색

3.2.1 깊이우선/너비우선 탐색

- 처음 두 가지 방법은 비용을 생각하지 않고 무작정 전체 경로를 순서대로 탐색하는 방법.
  - ❖ **깊이우선 탐색(Depth-first Search)**: 후보 노드들 중 하나의 노드를 전개하고 그 노드로부터 점점 다음 노드로 진행하는 것.
  - ❖ **너비우선 탐색(Breath-first Search)**: 매번 모든 후보 노드들을 전개하면서 진행하는 것.

- **깊이우선**은 진행하는 **방향이 맞으면 빠르지만**, 밑으로 더 진행하지 못하게 되면 되돌림이 필요하게 됨
- **너비우선**은 전체적으로 되돌림이 없어도 모두 전개하면서 진행하기 때문에 **속도가 느림.**

3.2.2 분기 한정법

- **분기 한정법(Branch and Bound Method)**은 깊이우선/너비우선 방법처럼 단지 도달하기만 하면 되는 것이 아니라 **가장 좋은 경로로 진행**.
- 좋은 경로, 즉 비용이 적은 경로를 선택하려면 비용을 정의하고 그것을 매번 평가하면서 진행.
- 비용의 정의는 평가 함수에 따라 이루어지지만 **분기 한정법에서는 누적 비용만 사용**.
- 탐색 순서는 깊이우선/너비우선 방법처럼 정해진 순서로 하지 않고 **평가 함수에 따라 누적 비용이 적은 경로를 우선적으로 진행**.
- 그러므로 어떤 상태에 있어서도 적어도 거기까지의 경로 중에서는 가장 좋은 경로로 진행하게 되고, 최종적으로 목표 상태에 도달했을 때는 전체 경로 중에서 가장 좋은 경로, 즉 **최적 경로를 구하게 됨**.
- 분기 한정법은 체계적으로 최적 경로를 구하는 가장 확실한 방법이지만, 여기저기 건너뛰어 돌아다니므로 현실적이지는 못함.

3.3 휴리스틱 탐색

- 누적비용 외에 **장래비용도 예측하여 탐색 효율을 높이는 것**.
- 여기서는 구체적인 비용 예측 방법에 대해서는 설명하지 않지만 어떤 경험치, 기대치, 보수 등이 있다는 것을 전제로 함.
- 이런 탐색법을 **휴리스틱 탐색(Heuristic Search)**이라고 하며 다음과 같은 종류가 있음.
  - **언덕 등반 탐색:** 장래 비용이 적은 방향으로 진행. 되돌림은 없음. 최적해 보장 없음. 평가함수: **f(n) = h(n)**
  - **최고우선 탐색:** 장래 비용이 적은 방향으로 진행. 다른 상태도 확인하면서 진행. 최적해 보장. 평가 함수: **f(n) = h(n)**
  - **A 알고리즘:** 과거의 경로를 기억하고 다른 상태도 확인하면서 진행. 최적 경로 보장. 평가함수: **f(n) = e(n) + h(n)**
  - **A* 알고리즘:** 장래 비용에 제약을 설정하여 A 알고리즘을 안정화. 최적 경로 보장. 평가 함수: **f(n) = e(n) + h(n)**

3.3.1 언덕 등반 탐색

- 평가 함수로서 장래 비용만을 사용하며, 과거 경로를 기억하지 않고 앞만 내다보고 진행. **등산하는 것처럼 정상만을 바라보고 오르는 것과 같음**.
- 장래 비용이 각 상태에서 잘 설정되어 있으면 최적해에 가장 빨리 도달할 수 있지만 장래 비용은 추측 값이기 때문에 경로를 그르쳐서 **국소해에 빠질 수도 있음**.
- 과거의 경로를 기억하지 않으므로 **되돌림이 불가능하며 결국에는 탐색 실패가 될 가능성도 있음**.
- 위험한 방법인 것처럼 보이지만 실제로는 우리들이 보통 사용하고 있는 방법임.
- 장래 비용은 커녕, 한 단계 앞만 보고 가장 좋은 방향으로 진행하는 방법도 있음. 그래도 한정된 시간, 판단 기준 내에서 진행하기에는 유효한 방법임.

3.3.1 언덕 등반 탐색

언덕 등반 탐색
(아래 그림에서 h를 고려)

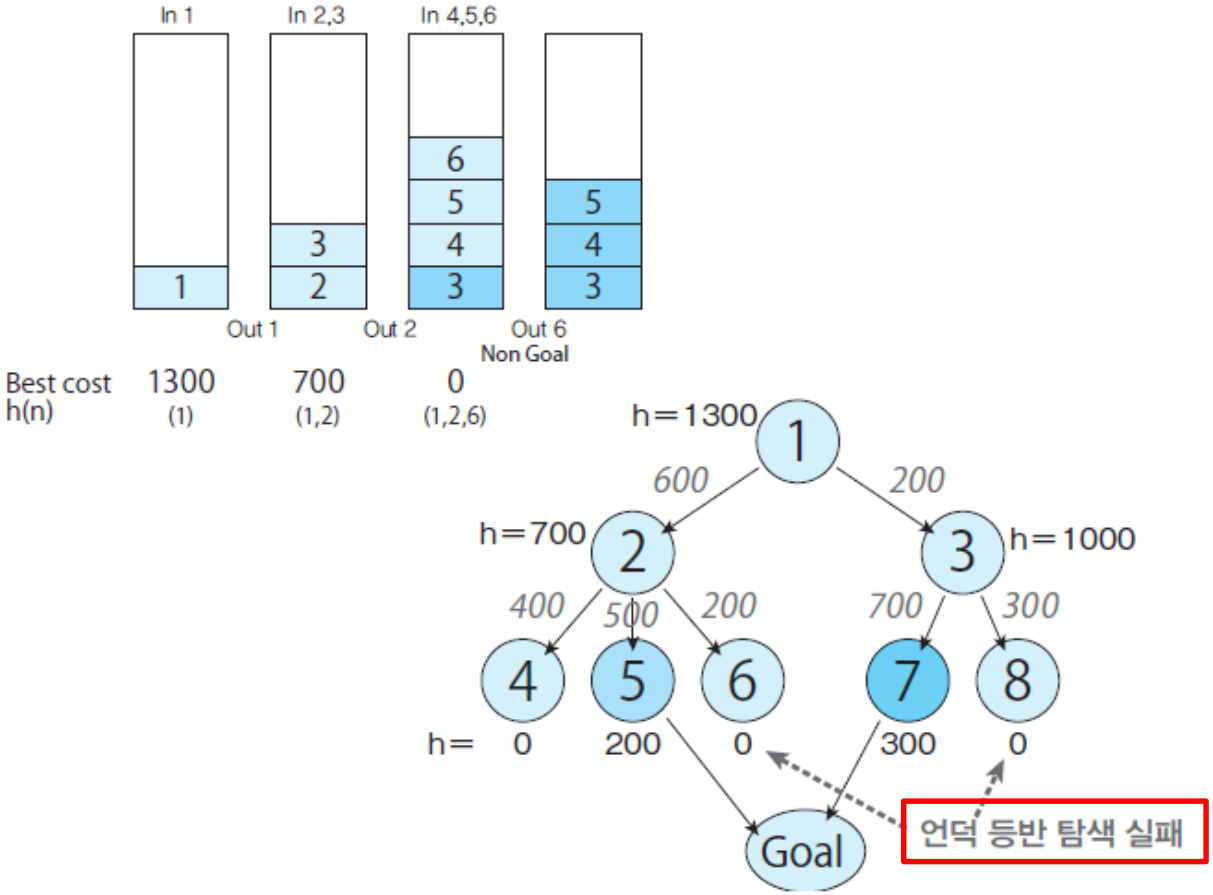


그림 6-3 언덕 등반 탐색(Hill-climbing Search)

3.3.2 최고우선 탐색

- 언덕 등반 탐색에서는 되돌림이 불가능하기 때문에 실패하는 경우가 있으므로, **국소해에 빠졌을 때 다른 상태를 다시 조사하도록 개선한 방법**.
- 평가 함수는 언덕 등반 탐색과 같이 장래 비용만 사용하며 과거 경로는 기억하지 않지만, **개선점으로 각 상태에 있어 아직 조사하지 않은 상태도 포함하여 평가 함수를 비교**.
- 만약 진행하고 있는 경로상에서의 장래 비용보다 조사하지 않은 상태(선택하지 않은 상태)에 대한 장래 비용 쪽이 적으면 현재 경로를 일단 중단하고 장래 비용이 가장 적은 다른 상태부터 다시 진행.
- 이렇게 함으로써 장래 비용만을 고려하고 있음에도 불구하고 어떤 상태에서도 적어도 그 시점에서는 가장 좋은 경로를 선택할 수 있음.
- 또한, **국소해에 빠져도 조사하지 않은 상태 중에서 장래 비용이 가장 적은 상태부터 다시 시작하는 것이 가능하므로 반드시 목표 상태에 도달**.

3.3.2 최고우선 탐색

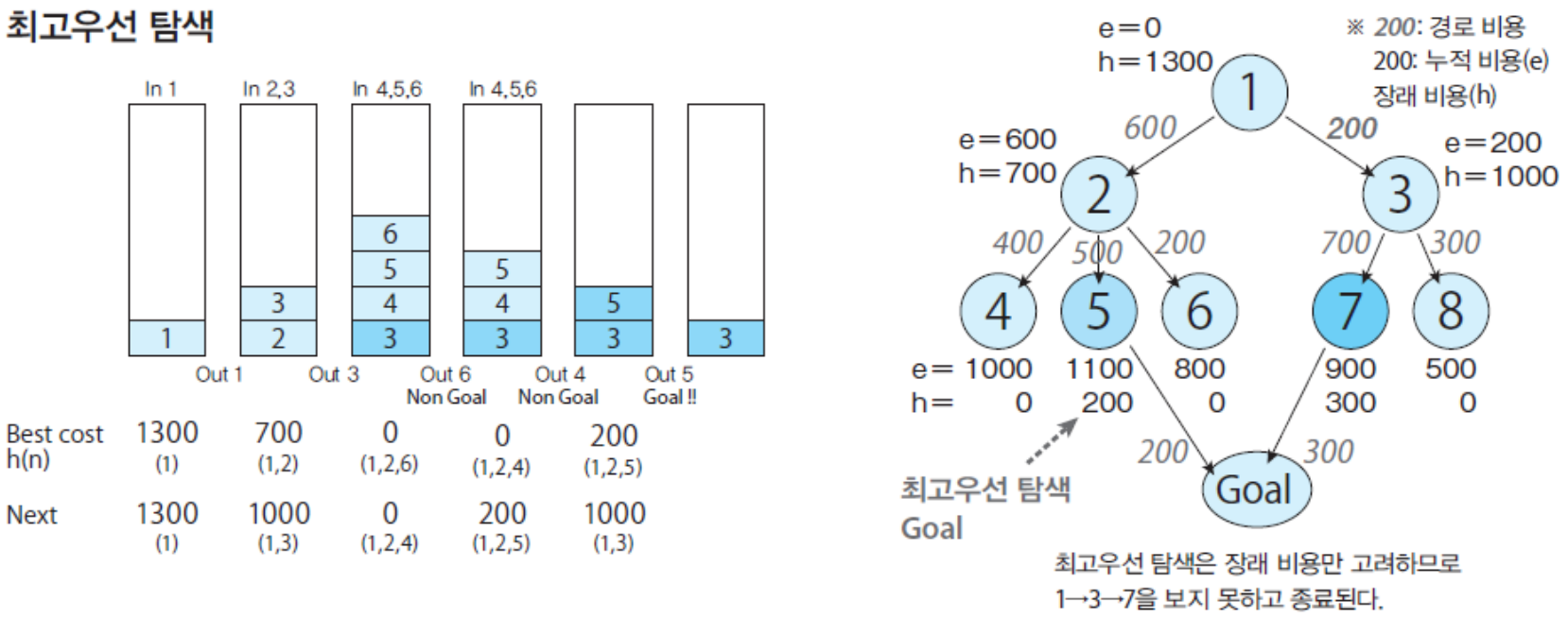


그림 6-4 최고우선 탐색(Best-first Search)

3.3.2 최고우선 탐색

- 언덕 등반 탐색의 장점인 속도를 살리면서도 분기 한정법에서 수행한 것처럼 조사하지 않은 상태도 고려하므로 **최적해를 보장할 수 있지만, 목표 상태에 도달한 시점에서 비로소 종료됨**.
- 즉, 일단 목표 상태에 도달하면 다른 상태는 조사하지 않으므로 **최적해이기는 하지만 최적 경로라는 보장은 없음**.
- **최고우선 탐색(Best-first Search)**은 **목표 상태에 반드시 도달하는 것을 목표**로 하고 있기 때문에 평가 함수에 누적 비용이 포함되지 않으며 과거 경로를 기억하지도 않음.
- 원래는 목표 상태에 도달하기까지 필요한 비용도 최소로 하고 싶기 때문에 또 다른 연구가 필요함.

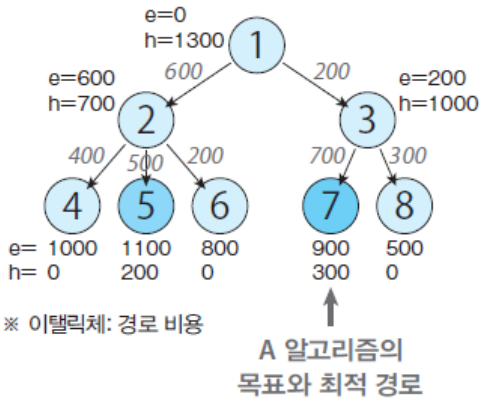
3.3.3 A 알고리즘

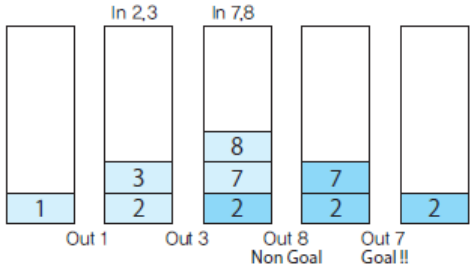
- **A 알고리즘(A Algorithm)**에서는 평가 함수로서 앞에서 설명한 식 6-1과 같이 **누적 비용과 장래 비용의 합계를 사용**.
- 과거 경로를 기억해 두고 각 상태에 대해 조사하지 않은 모든 상태들도 포함하여 평가 함수를 비교하면서 가장 값이 적은 방향으로 진행.
- 그러므로 **어떤 상태에 있어서도 항상 가장 좋은 경로를 선택하기 때문에 반드시 목표 상태에 최적 경로로 도달할 수 있음**.
- 여기저기서 경로를 중단하고 바꿔 가며 진행한다는 어려운 점은 있지만, 장래 비용을 추가한 것 때문에 분기 한정법으로 수행했던 것처럼 모든 경로를 건너뛰어 다니는 것은 아니므로 현실적임.

3.3.3 A 알고리즘

A 알고리즘

f(n)=e(n)+h(n) 단, e(n): 노드 n까지의 누적 비용, h(n): 노드 n부터 목표까지의 예측 비용

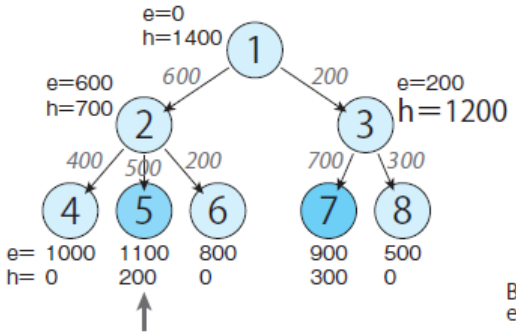


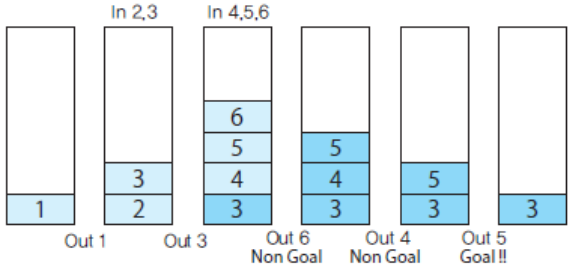


Best cost e(n)+h(n)	1300 (1)	1200 (1,3)	500 (1,3,8)	1200 (1,3,7)
Next	1300 (1)	1300 (1,2)	1200 (1,3,7)	1300 (1,2)

A 알고리즘은 평가 함수가 과거+장래이므로, 최적 경로를 보장할 수 있다. 단, h의 설정이 어렵다. 일반적인 기준으로 자식 노드 이후의 비용 중 최댓값을 설정한다. 과대한 값을 설정하지 않는다. h로서 한 단계 다음까지의 비용 밖에 보지 못한다면 분기 한정법과 같아진다.

h(n)을 과대하게 설정하면? 예를 들어, ❸에서 h=1200이라고 하면?





Best cost e(n)+h(n)	1400 (1)	1300 (1,2)	800 (1,2,6)	1000 (1,2,4)	1300 (1,2,5)
Next	1400 (1)	1400 (1,3)	1000 (1,2,4)	1300 (1,2,5)	1400 (1,3)

❸을 전개하지 못하고 최적이 아닌 경로로 목표 도달

A* 알고리즘
h를 과대하게 설정하지 않도록 제한
$h \leq h^*$: 진정한 장래 비용

그림 6-5 A 알고리즘과 고찰

3.3.4 A* 알고리즘

- **A* 알고리즘(A* Algorithm)**에서는 A 알고리즘의 장래 비용에 **다음과 같은 제약 조건을 설정**.

식 6-2

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

$h(n) \leqq h^*(n)$

$h^*(n)$ … n부터 장래의 진정한 최소 비용*4

- 장래 비용에 위의 식과 같은 제약을 추가하면 어떤 상태에서도 거기서부터 앞으로의 진정한 비용 이하의 상태가 선택되는 것이기 때문에 목표 상태에 도달했을 때 진정한 비용(이 시점에서는 누적 비용과 같은 값으로 확정값)을 유지하는 것이 가능.
- A* 알고리즘에 의해 **최적해(목표 상태 도달) 및 최적 경로를 완전히 보장할 수 있음**.

3.4 탐색법 정리

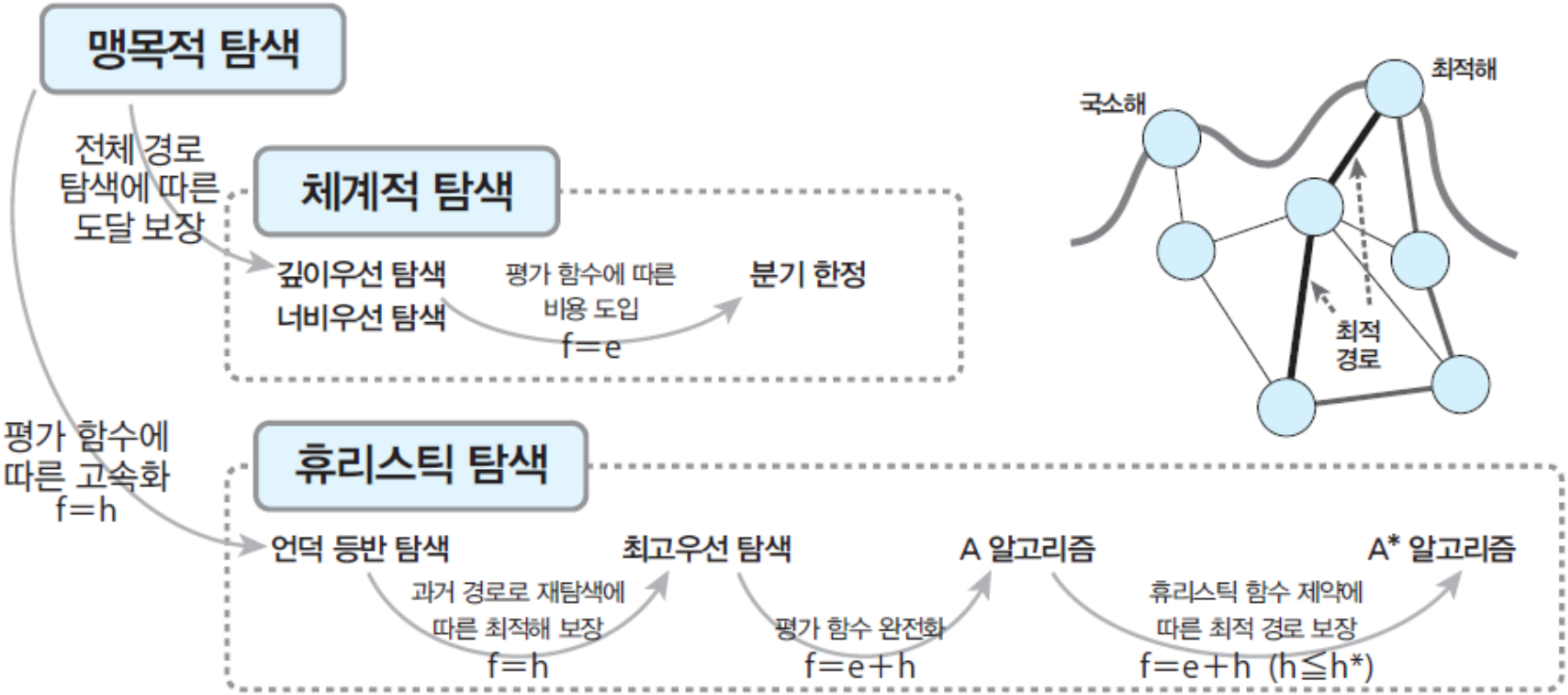
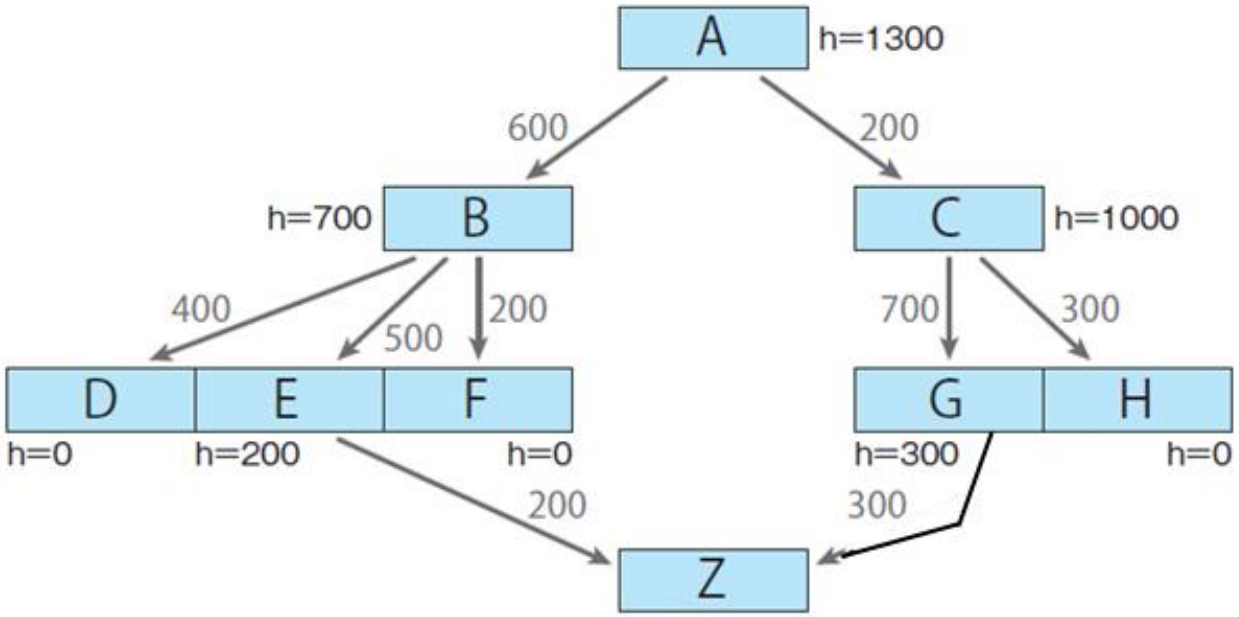


그림 6-6 탐색법 정리

⊙ 체험해 봅시다: 탐색법의 비교



시뮬레이션에서 사용되는
Default Tree
(A에서 Z로 이르는 경로)

- A~Z: 노드
- 경로상의 숫자: 경로 비용
- 노드상의 숫자(h): 해당 지점 부터 정상까지의 예측 비용

그림 6-1 시뮬레이션에 사용되는 탐색 트리