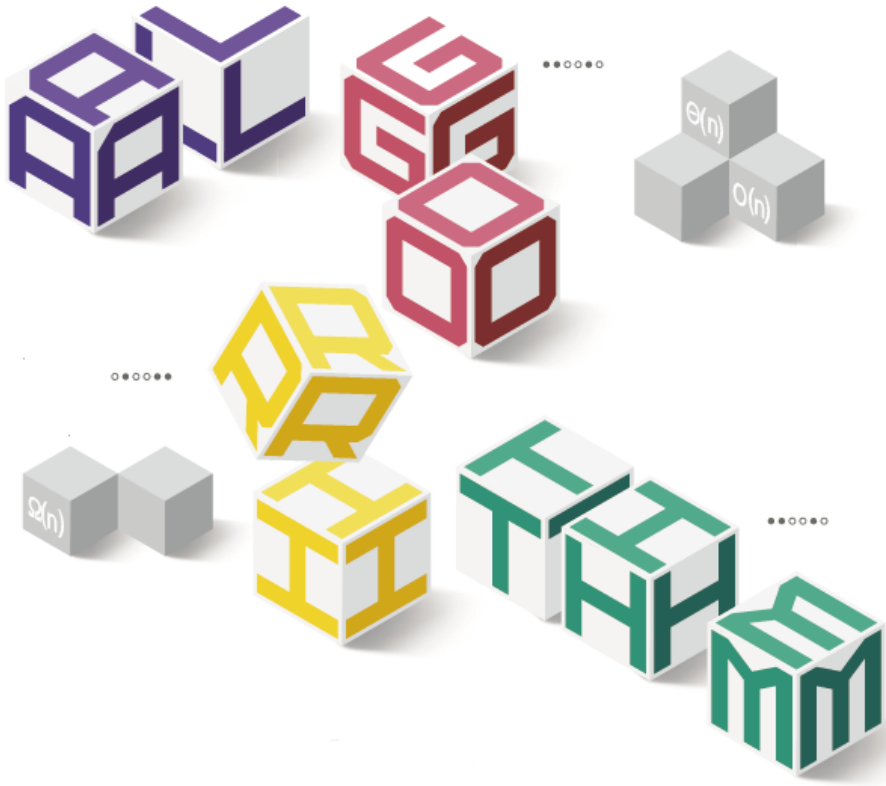
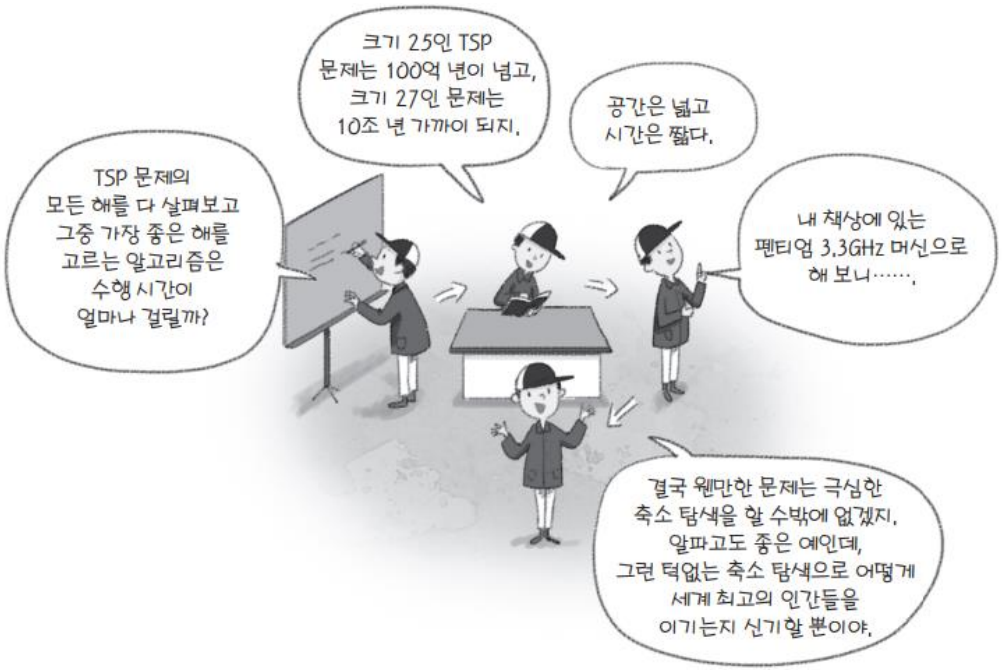


관계 중심의 사고법

쉽게 배우는 알고리즘

12장. 상태공간 트리의 탐색

12장. 상태공간 트리의 탐색



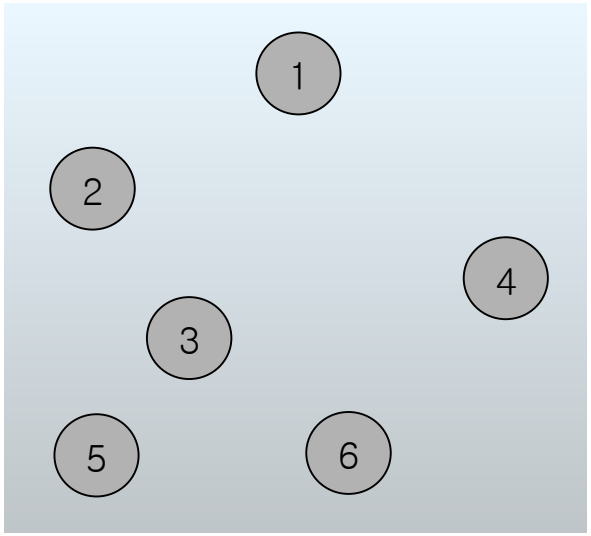
학습목표

- 상태 공간 트리의 탐색을 이해한다.
- 상태 공간 트리가 무엇인지 이해한다.
- 백트래킹 기법의 작동 원리를 이해한다.
- 한정 분기의 작동 원리를 이해하고, 백트래킹에 비해 장점이 무엇인지 이해하도록 한다.
- A* 알고리즘의 작동 원리를 이해하고, 어떤 문제들이 A* 알고리즘의 적용 대상인지 감지하도록 한다.

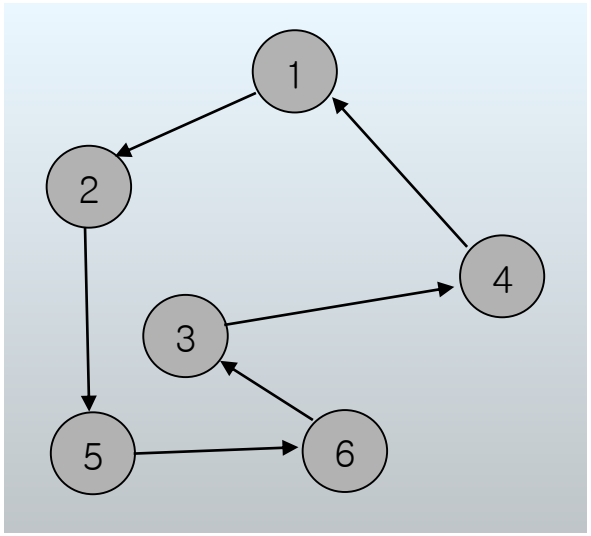
상태공간트리State-Space Tree

- 문제 해결 과정의 중간 상태를 각각 한 노드로 나타낸 트리
- 이 장에서 배우는 세가지 상태공간 탐색 기법
  - 백트래킹
  - 분기한정
  - A* 알고리즘

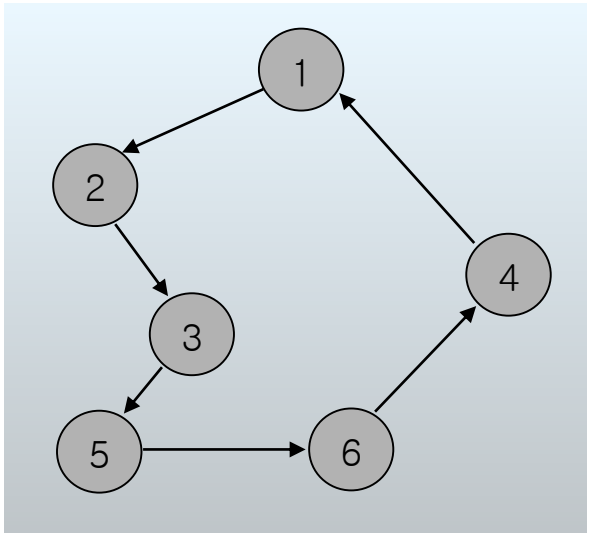
TSP의 예



(a) TSP 예제



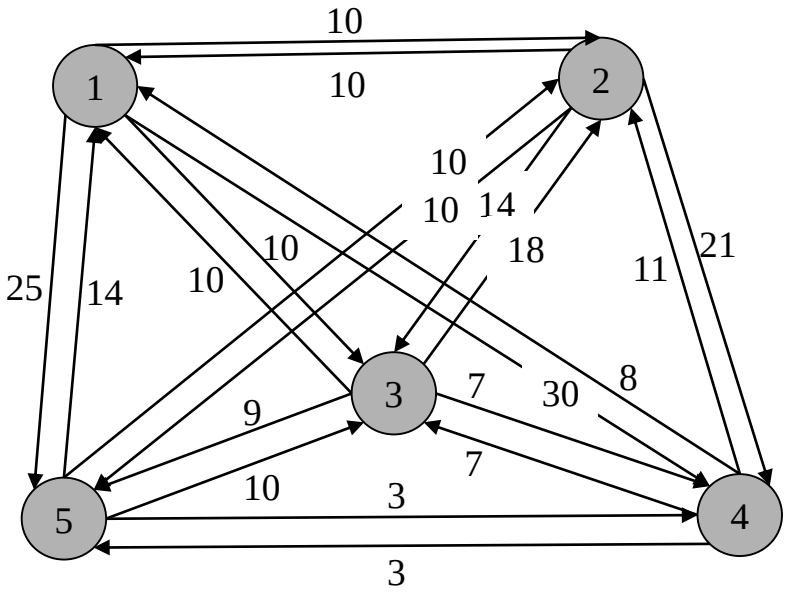
(b) 해의 예



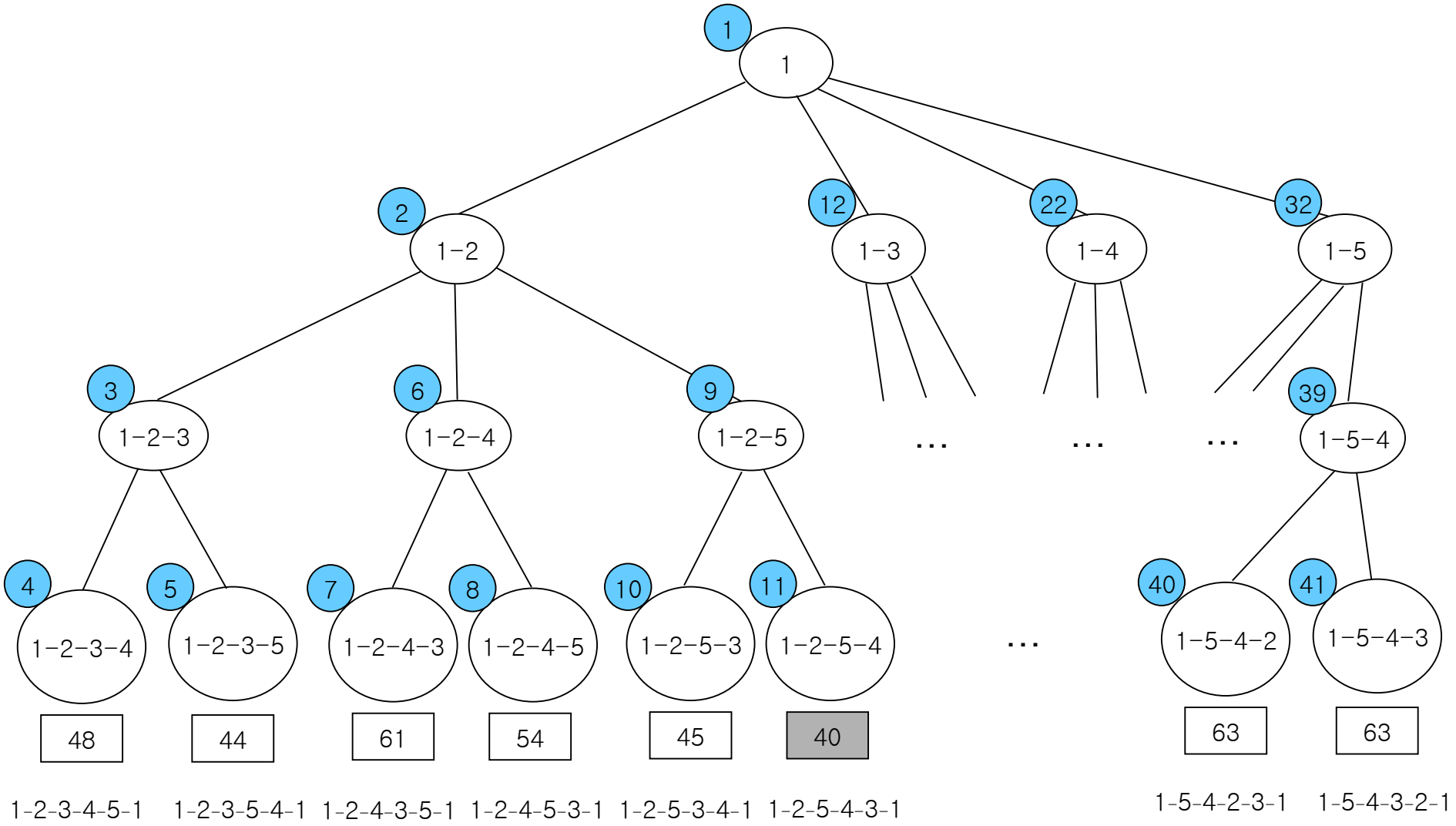
(c) 최적해

TSP와 인접행렬의 예

	1	2	3	4	5
1	0	10	10	30	25
2	10	0	14	21	10
3	10	18	0	7	9
4	8	11	7	0	3
5	14	10	10	3	0



사전적 탐색의 상태공간트리

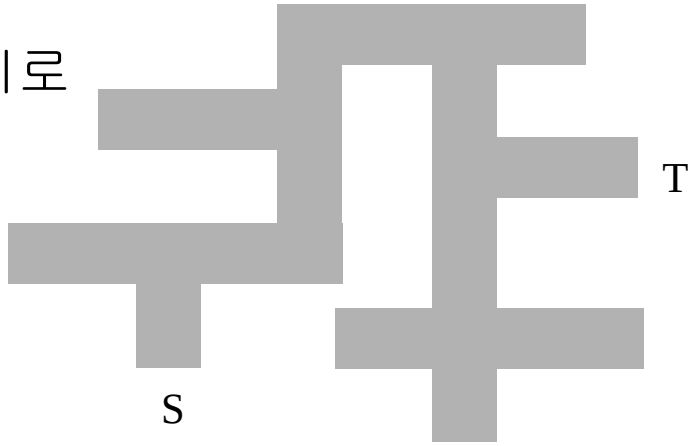


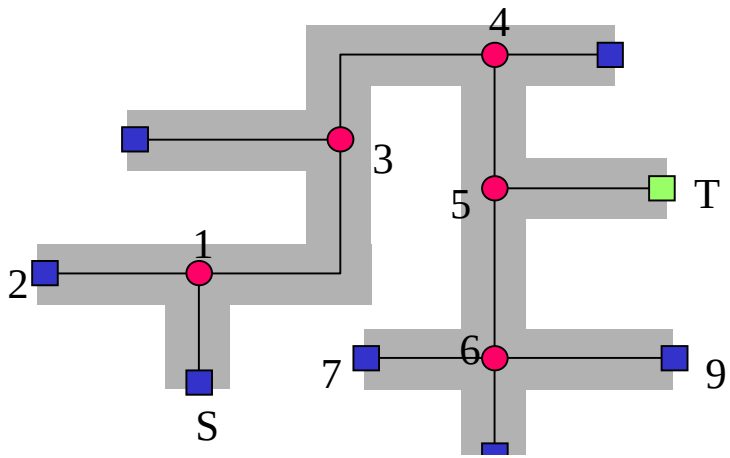
백트래킹

- DFS 또는 그와 같은 스타일의 탐색을 총칭한다
- Go as deeply as possible, backtrack if impossible
  - 가능한 지점까지 탐색하다가 막히면 되돌아간다
- 예
  - 미로찾기, 8-Queens 문제, 지도 색칠하기, …

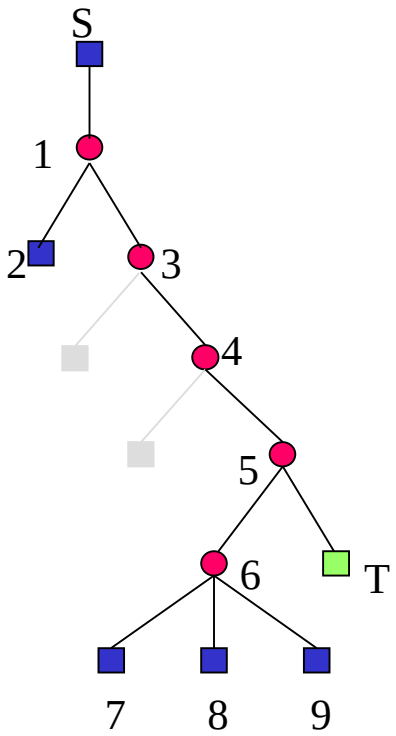
미로찾기 문제

(a)미로





(b)그래프로 모델링한 미로



(c)분기 트리

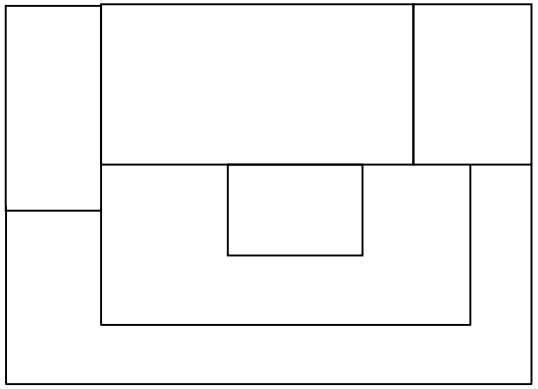
미로 찾기 문제를 위한 백트래킹 알고리즘

```
maze(v)
{
        visited[v] ← YES;
        if (v = T) then {print "성공!";}    ▷ 끝내기
        for each x ∈ L(v)                   ▷ L(v) : 정점 v의 인접 정점 집합
                if (visited[x] = NO) then {
                        prev[x] ← v;
                        maze(x);
                }
}
```

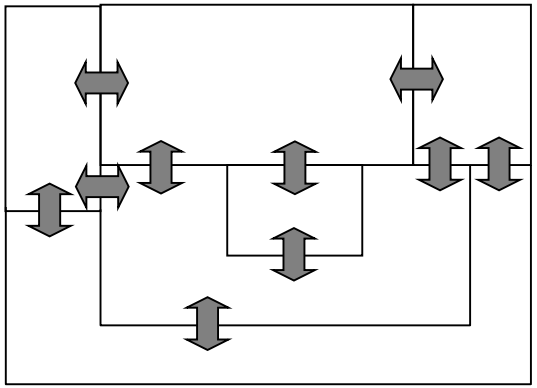
색칠 문제

- 그래프에서
  - 인접한 정점은 같은 색을 칠할 수 없다
  - k 개의 색상을 사용해서 전체 그래프를 칠할 수 있는가?

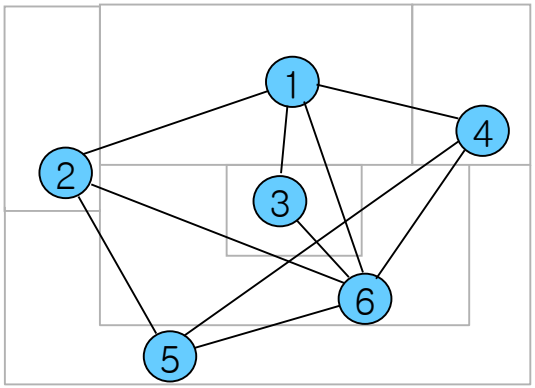
색칠 문제의 예: 지도 색칠



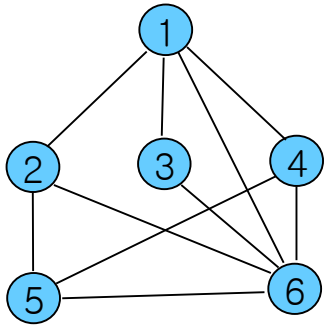
(a) 지도



(b) 구역간의 인접관계



(c) 연결관계를 정점과 간선으로 나타낸 것



(d) (c)와 동일한 그래프

```
kColoring(i , c)
▷ i: 정점, c: color
▷ 질문: 정점 i-1까지는 제대로 칠이 된 상태에서 정점 i를 색 c로 칠하려면
    k 개의 색으로 충분한가?
{
    if (valid(i, c)) then {
        color[i] ← c;
        if (i = n) then {return TRUE;}
        else {
            result ← FALSE;
            d ← 1;                                  ▷ d: color
            while (result = FALSE and d ≤ k) {
                result ← kColoring(i+1, d);   ▷ i+1: 다음 정점
                d++;
            }
        }
        return result;
    } else {return FALSE;}
}
```

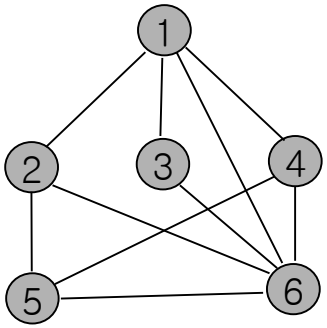
valid(i, c)

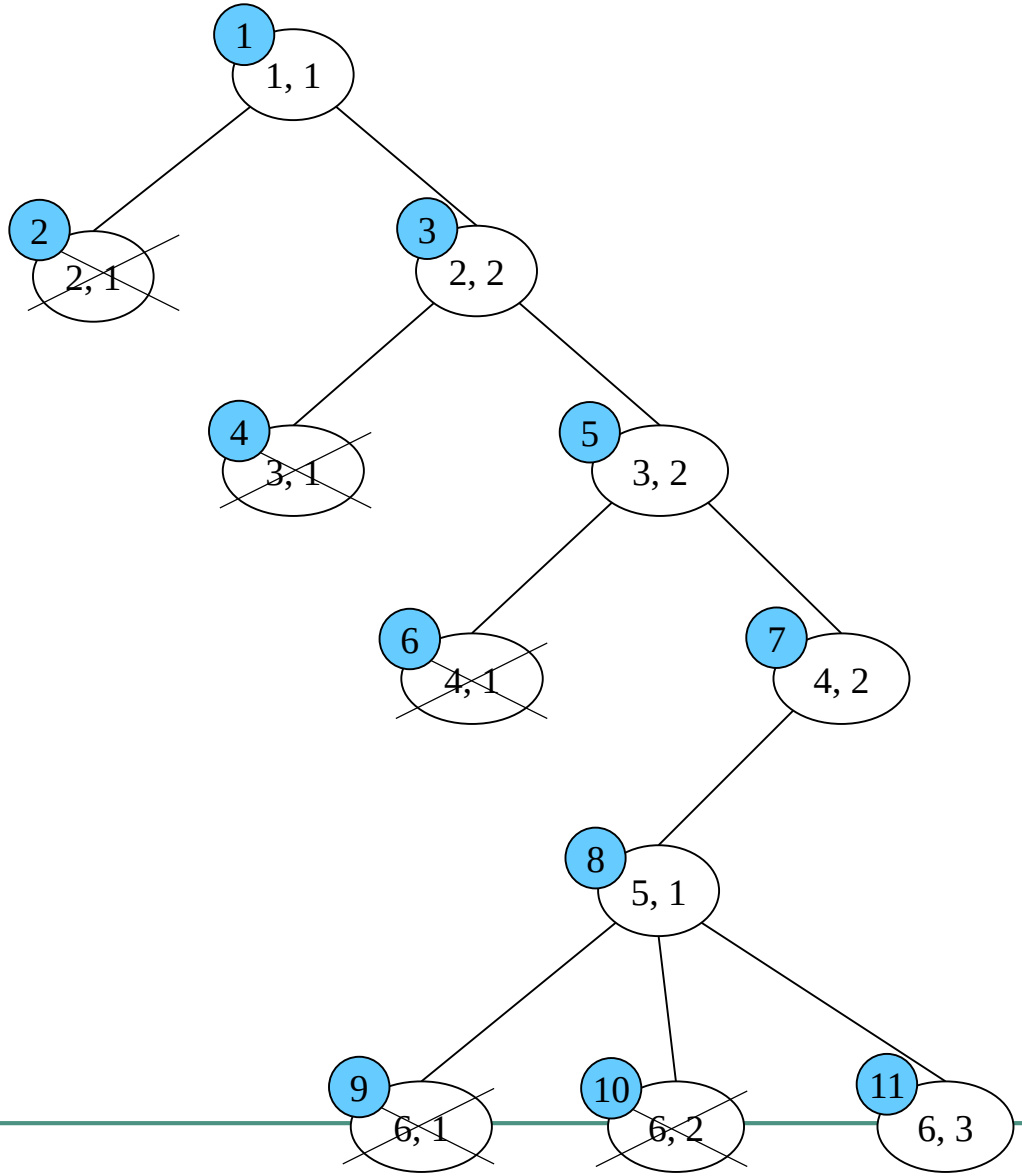
▷ i: 정점, c: color

▷ 질문: 정점 i-1까지는 제대로 칠이 된 상태에서 정점 i를 색 c로 칠하려면 이들과 색이 겹치지 않는가?

```
{
    for j ← 1 to i-1 {
            ▷ 정점 i와 j 사이에 간선이 있고, 두 정점이 같은 색상이면 안된다
            if ((i, j) ∈ E and color[j] = c) then return FALSE;
    }
    return TRUE;
}
```

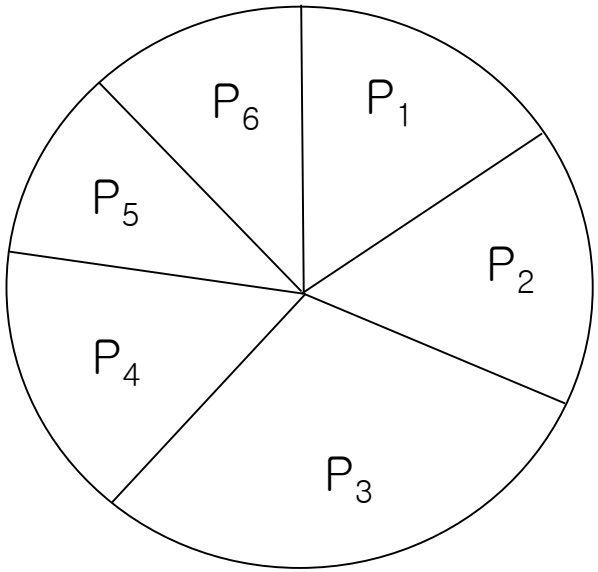
그래프 색칠 백트래킹 알고리즘의 상태공간트리



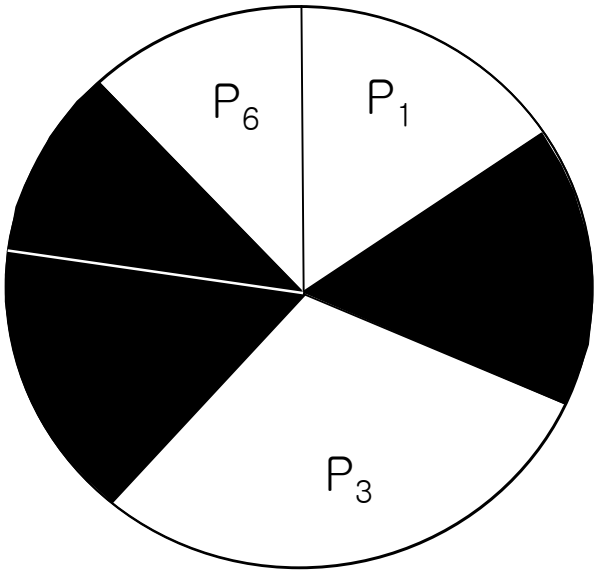


한정분기Branch-and-Bound

- 분기branch와 한정bound의 결합
  - 분기를 한정시켜 쓸데없는 시간 낭비를 줄이는 방법
- 백트래킹과 공통점, 차이점
  - 공통점
    - 경우들을 차례로 나열하는 방법 필요
  - 차이점
    - 백트래킹 – 가보고 더이상 진행이 되지 않으면 돌아온다
    - 분기한정 – 최적해를 찾을 가능성이 없으면 분기는 하지 않는다

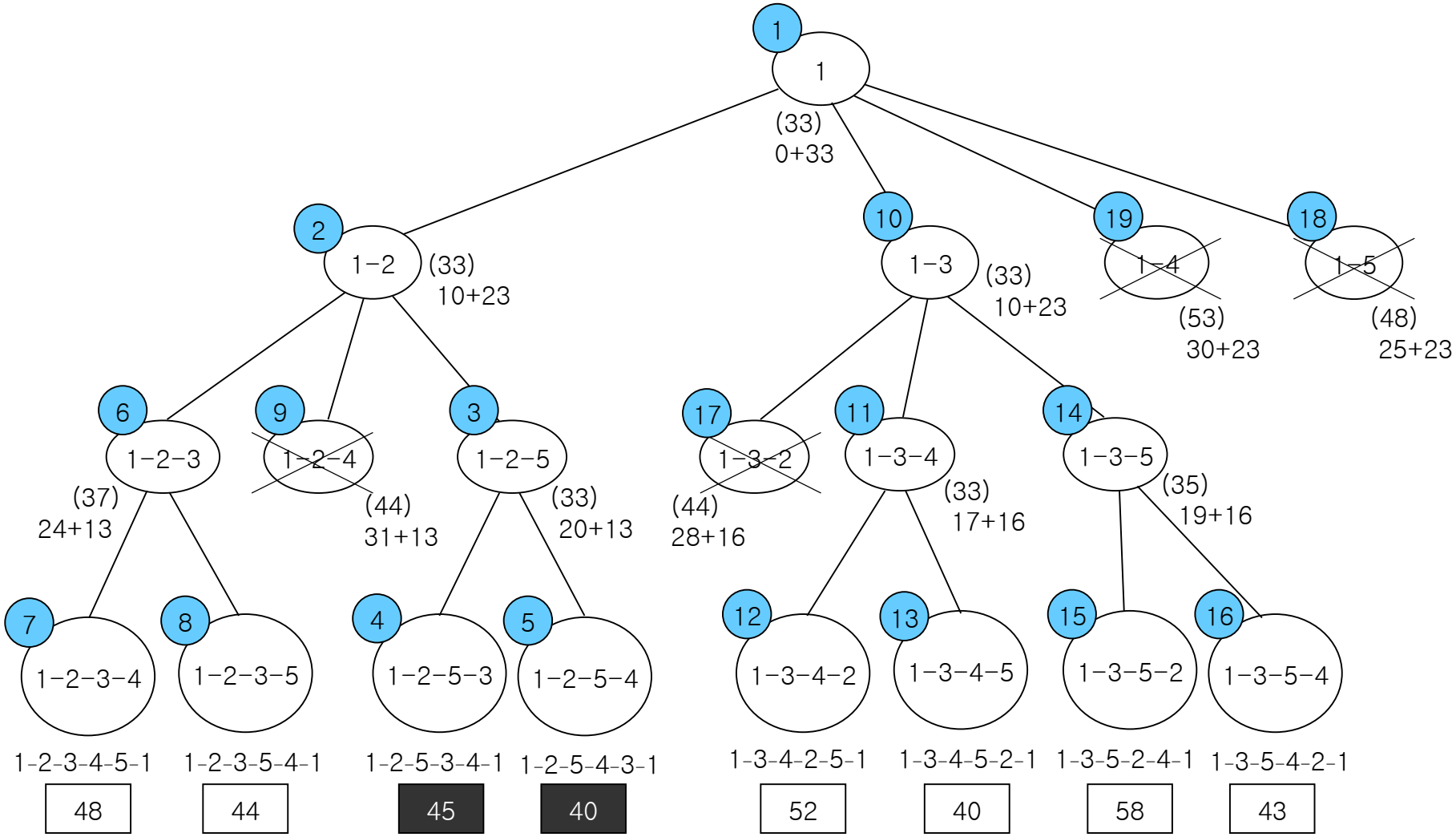


(a) 어느 시점에 가능한 선택들



(b) 최적해를 포함하지 않아 제외하는 선택들

TSP 예제를 대상으로 한 한정분기 탐색의 예 (상태공간트리)



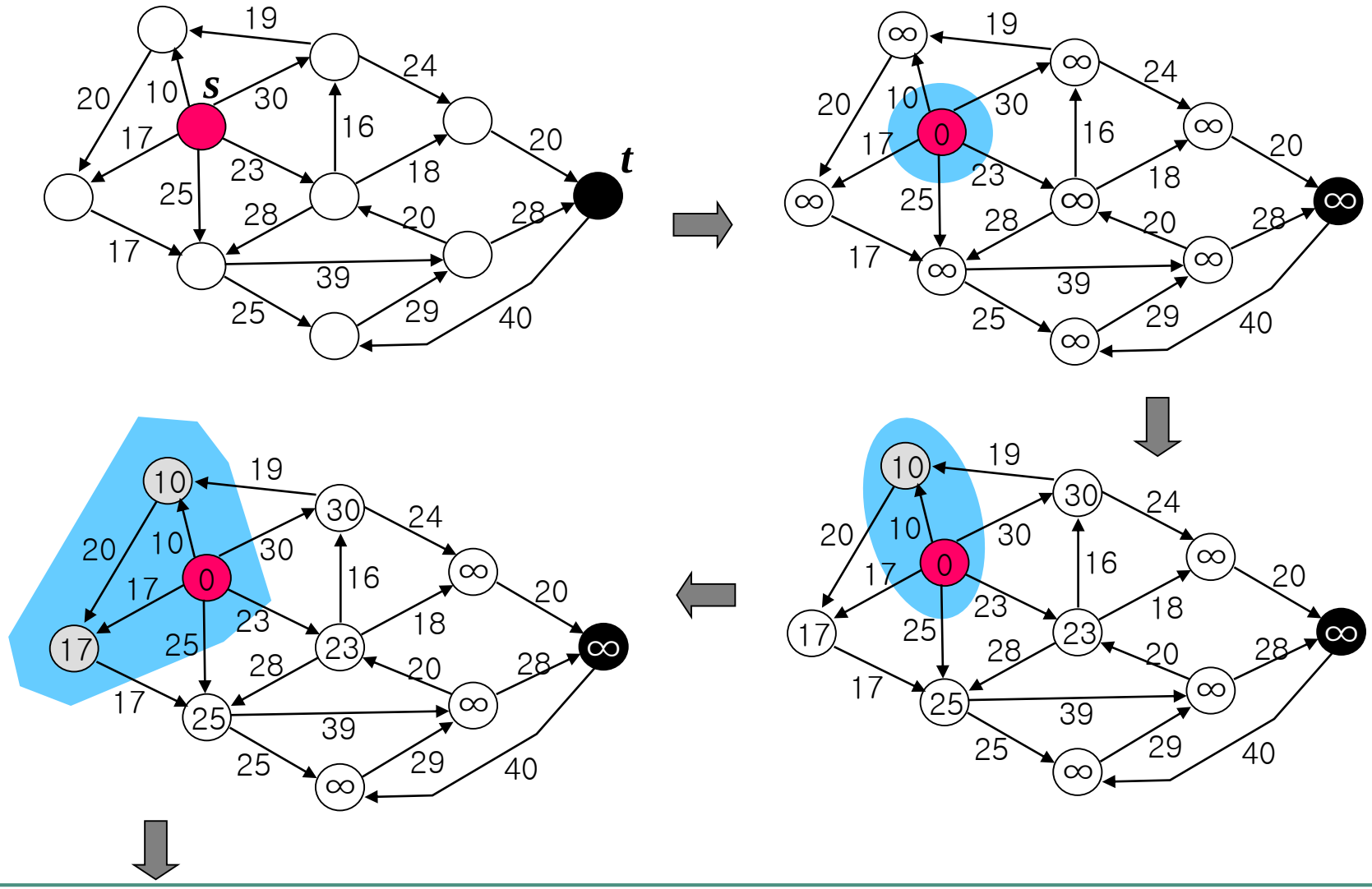
A* 알고리즘

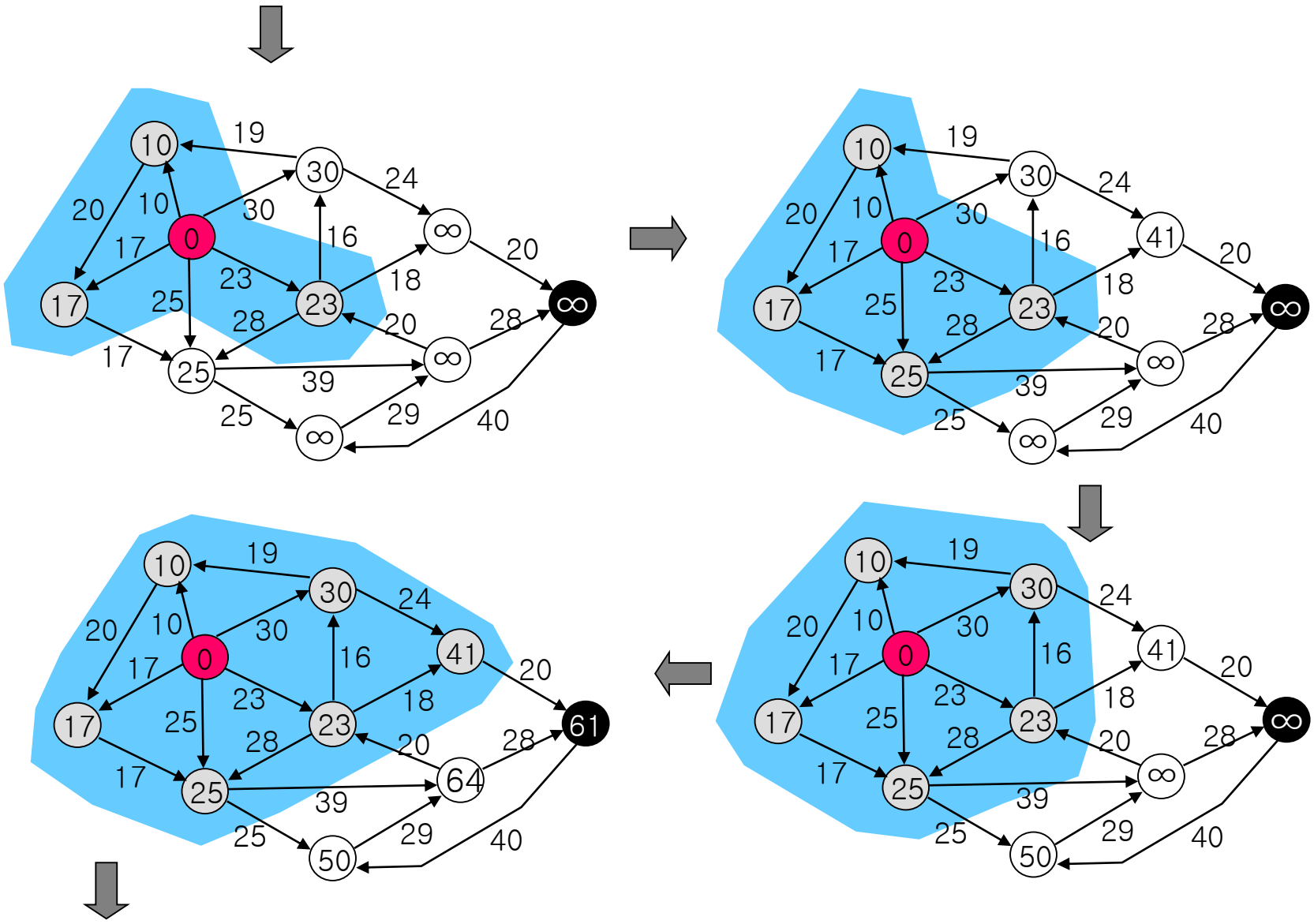
- 최적우선탐색
  - 각 정점이 매력함수값 $g(x)$를 갖고 있다
  - 방문하지 않은 정점들 중 $g(x)$ 값이 가장 매력적인 것부터 방문한다
- A* 알고리즘은 최적우선탐색에 목적점에 이르는 잔여추정거리를 고려하는 알고리즘이다
  - 정점 x로부터 목적점에 이르는 잔여거리의 추정치 $h(x)$는 실제치보다 크면 안된다

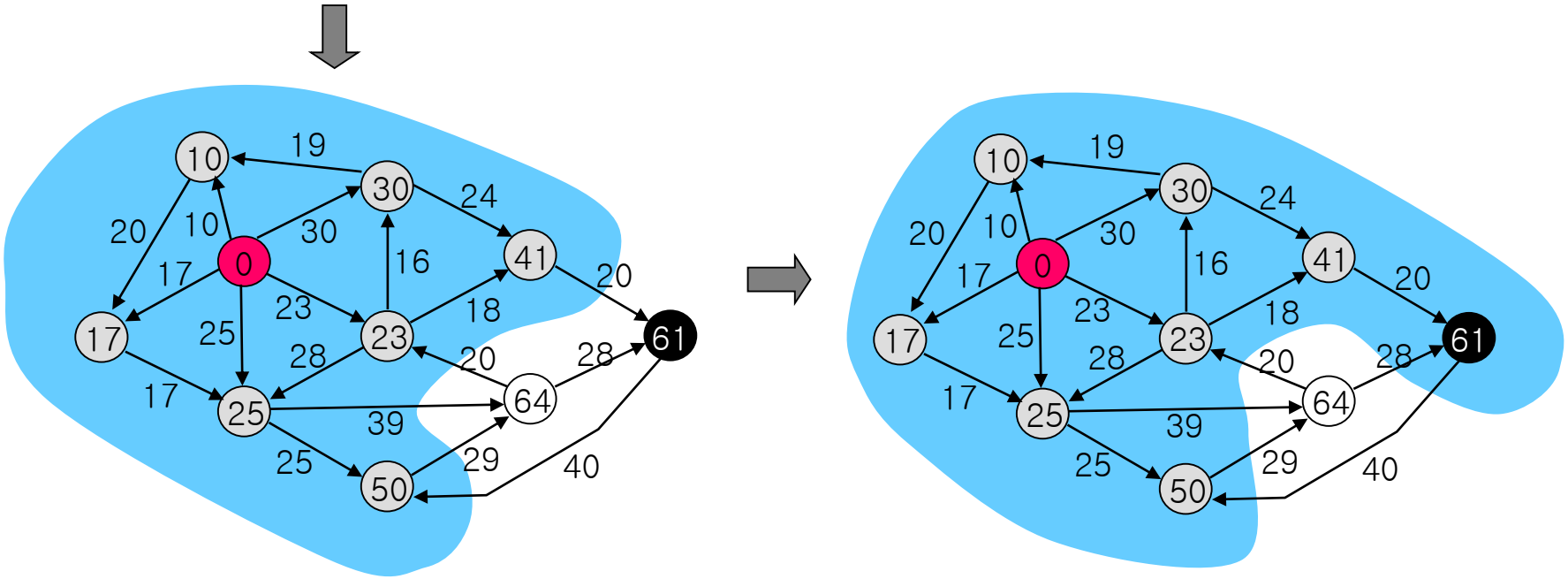
최단경로 문제

- Remind: 다익스트라 알고리즘
  - 시작점은 하나
  - 시작점으로부터 다른 모든 정점에 이르는 최단경로를 구한다 (목적점이 하나가 아니다)
- A* 알고리즘에서는 목적점이 하나다

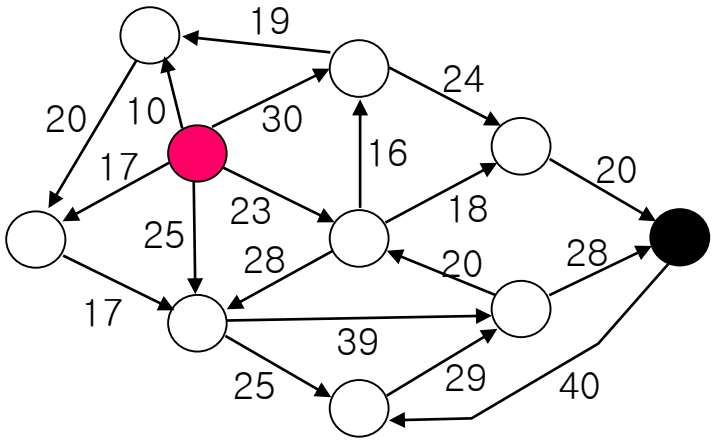
다익스트라Dijkstra 알고리즘의 작동 예

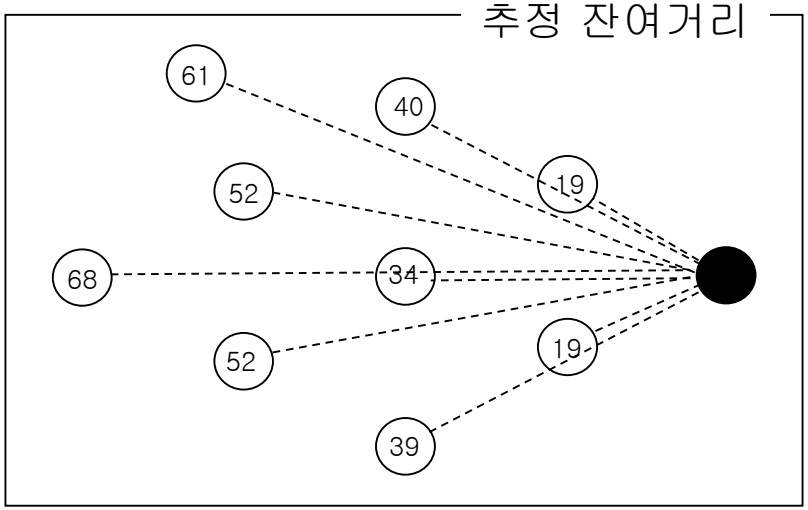


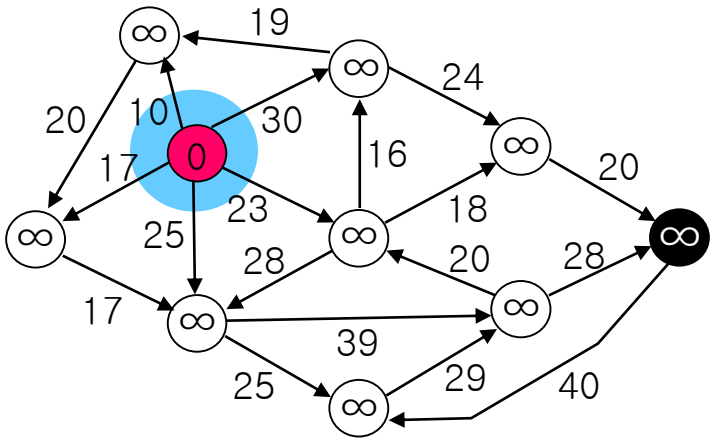

10
19
30
24
20
10
30
∞
16
20
17
0
23
18
17
25
28
23
20
28
∞
17
25
39
25
∞
29
40
41
61
64
50

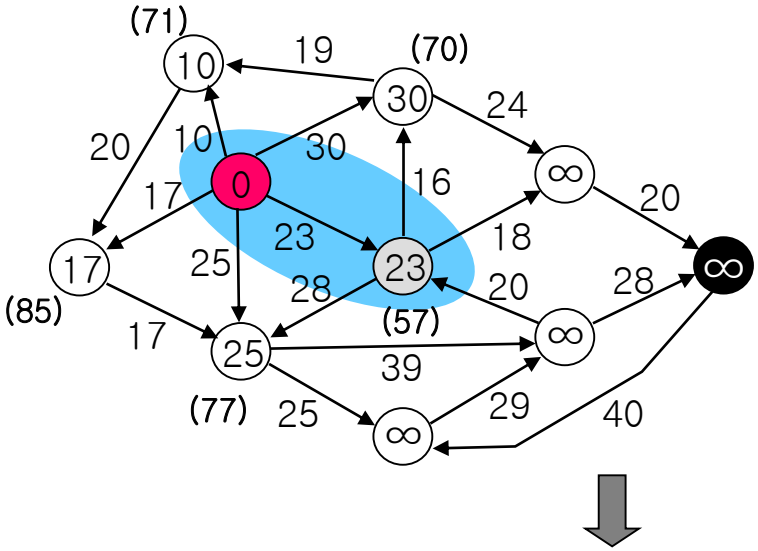

10
19
30
24
20
10
30
41
0
16
17
20
23
18
61
17
25
28
20
28
64
17
25
39
25
29
40
50

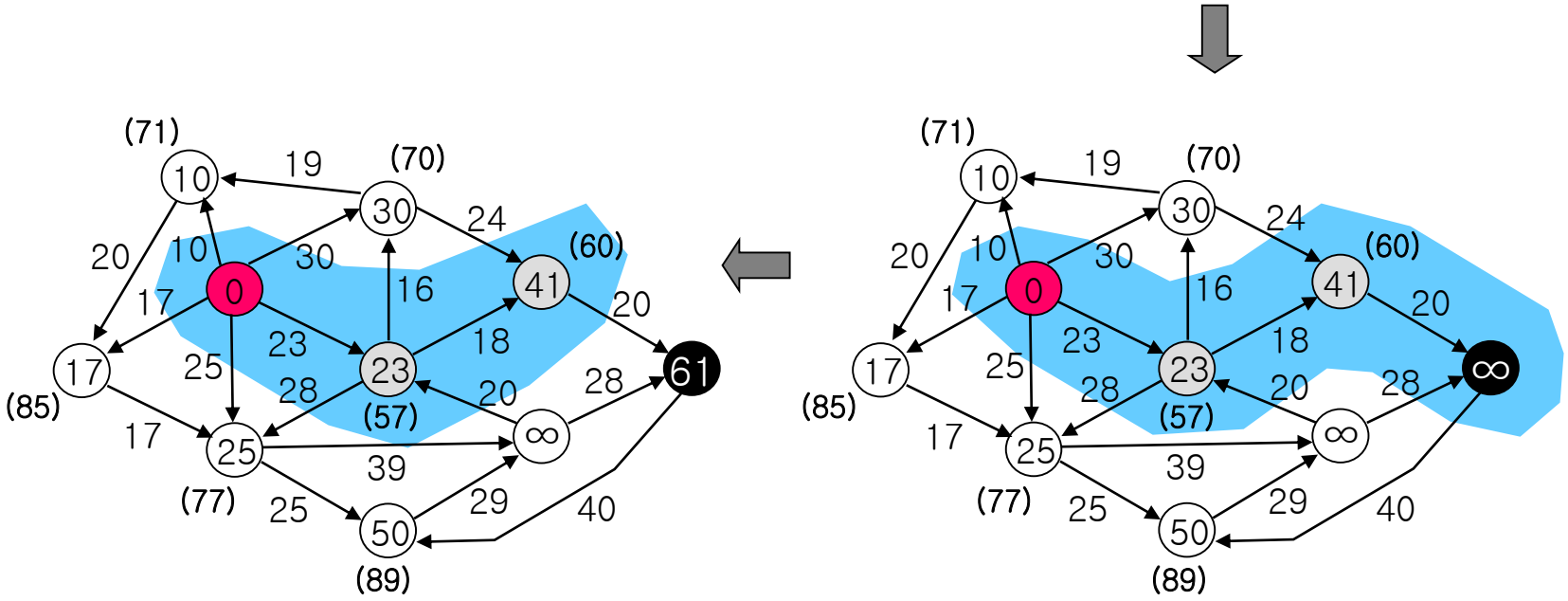
A* 알고리즘의 작동 예


19
24
20
10
30
17
16
20
23
18
25
28
20
28
17
39
25
29
40


추정 잔여거리
61
40
52
19
68
34
52
19
39


∞
19
∞
24
20
10
30
∞
17
0
16
20
23
18
∞
25
∞
28
20
28
∞
17
∞
∞
39
25
29
40
∞

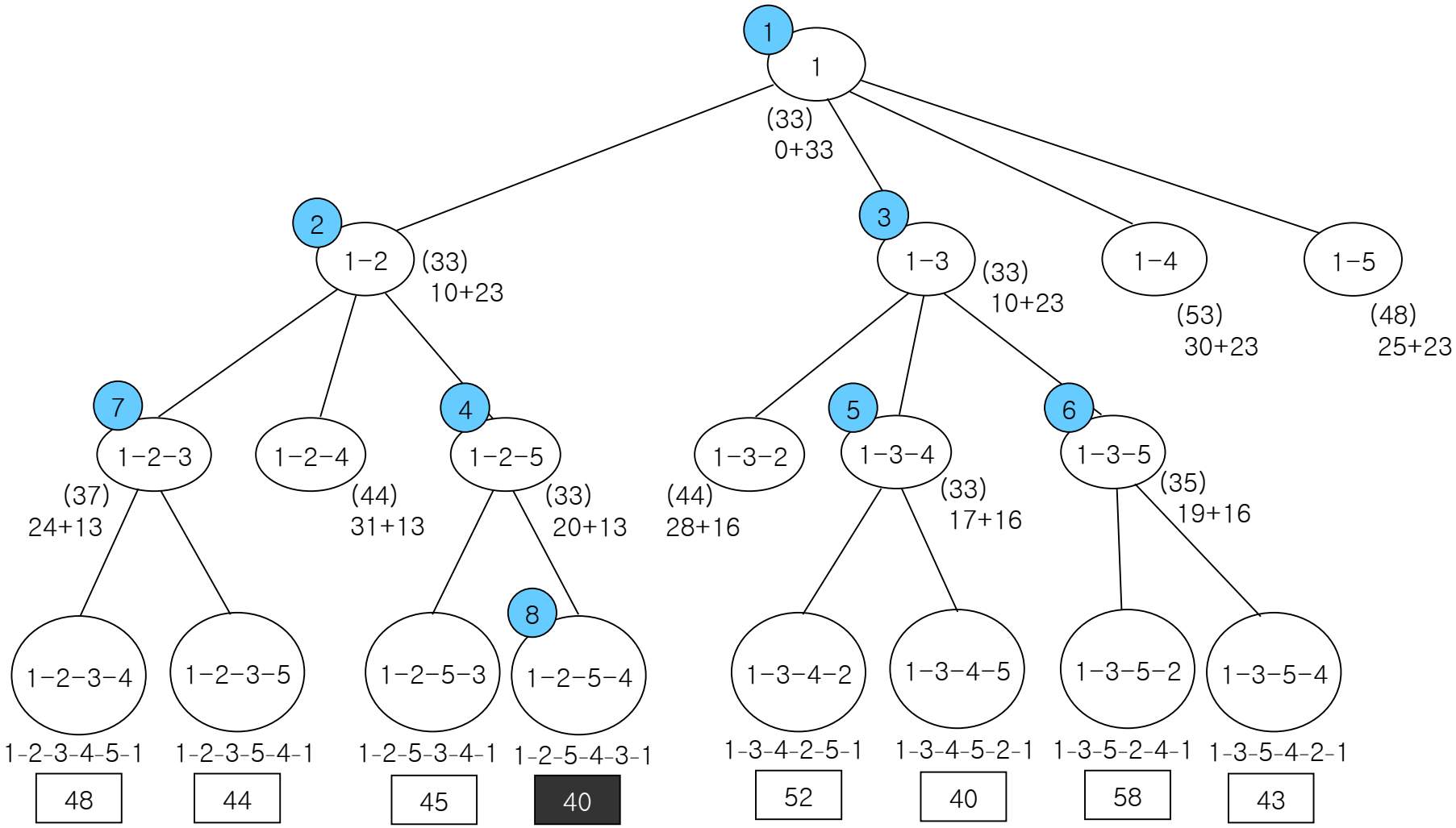

(71)
10
19
(70)
30
24
20
10
30
17
0
16
∞
20
23
18
17
25
23
∞
28
20
28
(85)
(57)
17
25
∞
39
(77)
25
∞
29
40



✓ 추정잔여거리를 사용함으로써 탐색의 단계가 현저히 줄었다

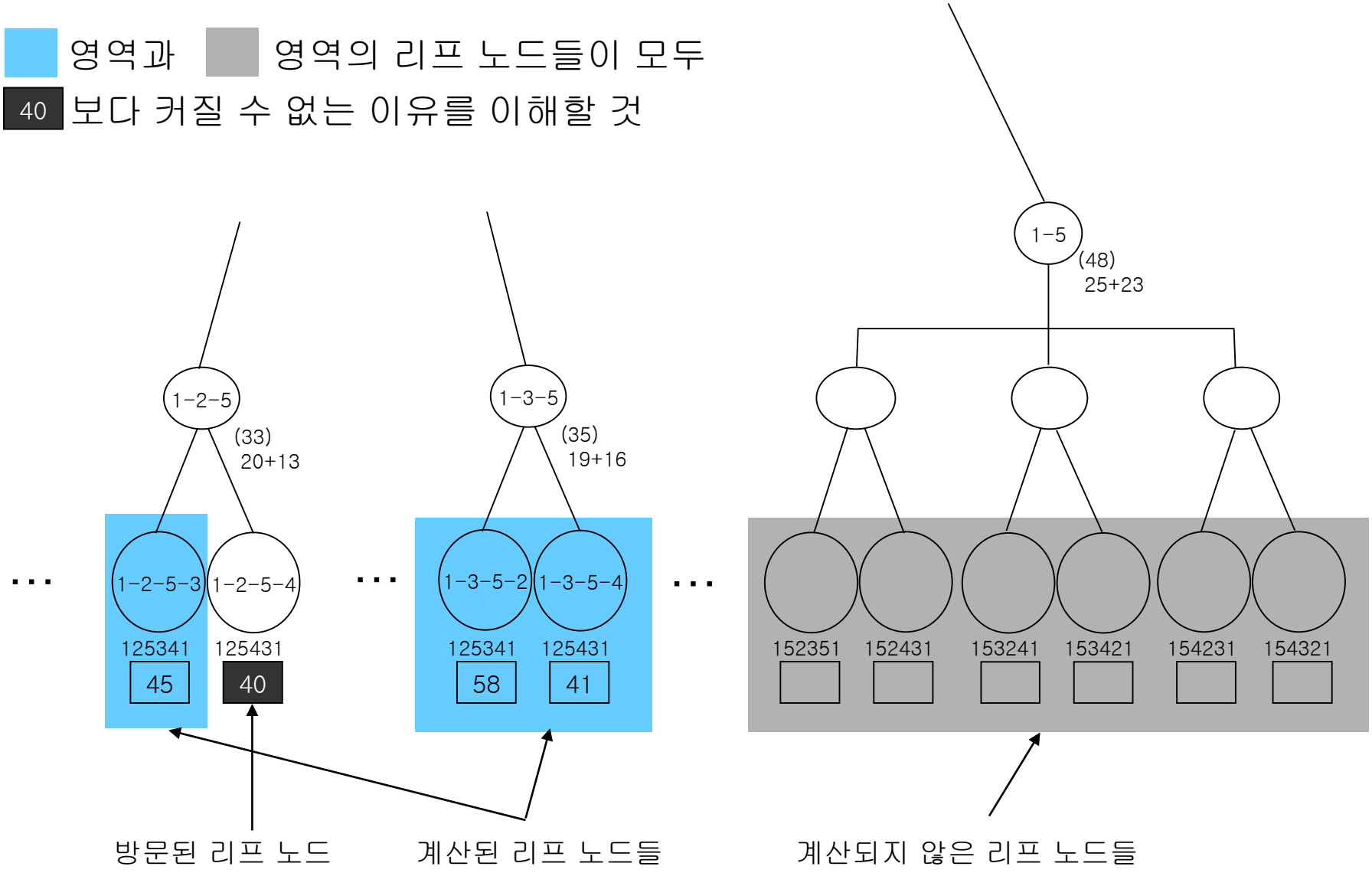
✓ 전제 조건: $\forall x, y \in V, h(x) \leq w(x,y) + h(y)$

TSP 예제를 대상으로 한 A* 알고리즘 탐색의 예 (상태공간트리)



A* 알고리즘이 첫 리프 노드를 방문하는 순간 종료되는 이유

영역과 영역의 리프 노드들이 모두 40 보다 커질 수 없는 이유를 이해할 것



Thank you