문제 해결 및 탐색 전략

Romania 예제

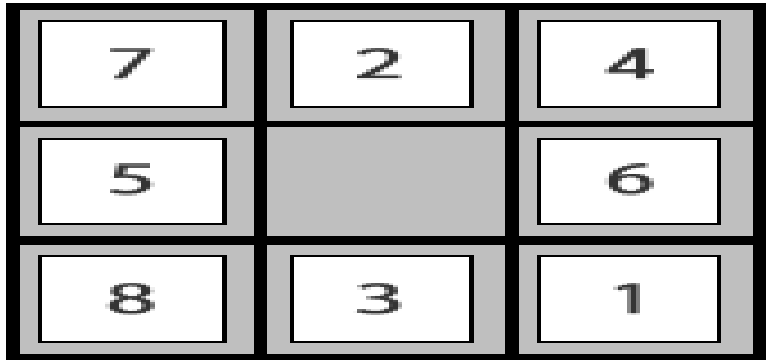
- 문제 정의 하기

- 초기상태 : Arde(현재 있는 도시)
- 목표 : Bucharest
- State : 각 도시들
- Action : state 이동
- Solution : Ared에서 Bucharest를 가는 경로(최적해, 최단거리)

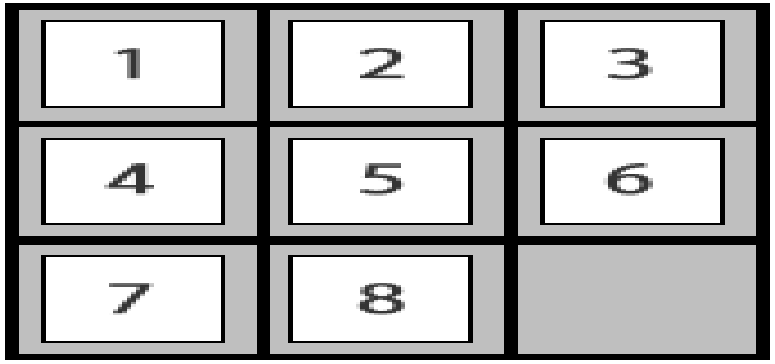
문제 정의 구성요소

- Initial state
- 초기 상태(Ared)
- Possible actions
- 가능한 행동들을 정의(Successor 함수 정의)
- Goal test
- GoId에 도달했는지 아닌지를 판단
- Path Cost
- 어떤 state에서 어떤 action을 취해 다른 state로 간다.(->경로, Path)
- 경로를 수행하는데 발생하는 비용(0보다 크거나 같다.)
- 두 도시 사이의 거리

The 8-puzzle 예제



Start State



Goal State

[Note: an optimal solution of n-Puzzle family is NP-hard]

- States? Locations of tiles
- Actions? Move blank left, right, up, down
- Goal test? Goal state (given)
- Path cost? 1 per move

Tree Search Algorithms

- State space(모든 가능한 state)를 다 펼치다 보면 Tree구조를 가짐
- Initial state : Tree의 Root
- 각 node : 도시이름(state)
- Action : 각각의 edge(Tree에서 branch 생성)
- 탐색 : State를 펼치고, 그 중에 어느 state로 가서 또 펼치는 과정을 반복하여 goal state를 찾는 과정

탐색 전략

- 각각의 노드를 펼칠 때 어떤 순서로 펼쳐야 하는지 정의
- Completeness(완전성)
- 내가 만약 이 전략을 따른다면 반드시 solution을 찾을 수 있는가?
- 답이 Yes라면 해를 찾을 수 있다는 것이 보증됨
- Time complexity(시간복잡도)
- 얼만큼의 노드를 펼칠 때 goal을 찾을 수 있는가?
- 낮을수록 좋은 전략
- Space complexity(공간복잡도)

-노드를 펼쳐 놓고 메모리 상에 저장을 해 놓아야함.

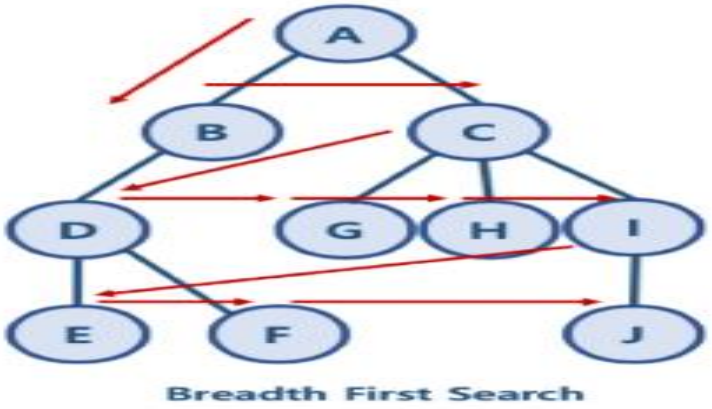
- 낮을수록 좋은 전략
- Optimality
- 항상 최적의 solution을 찾을 수 있는가?

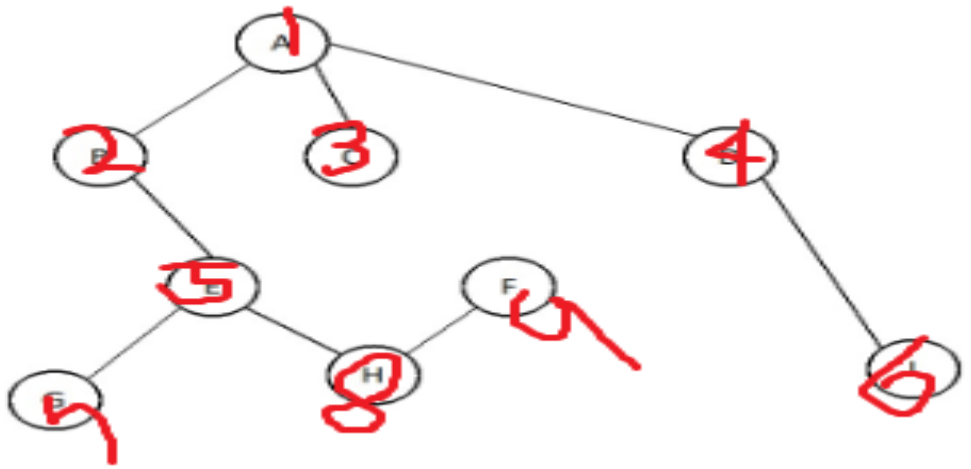
Uninformed search

- Breadth-first search(너비 우선 탐색)
- Depth-first search(깊이 우선 탐색)

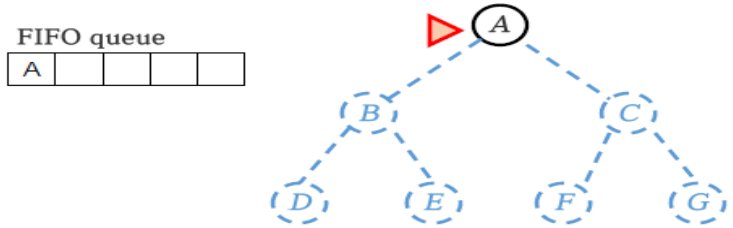
Breadth-first search(너비 우선 탐색)

- 만약에 주어진 노드가 있고 아직 펼쳐지지 않은 노드들이 있다면, 가장 얕은 (아직 펼쳐지지 않은) 노드부터 펼치자.
- "같은 level 노드들 먼저 순회"
- FIFO(first in first out) , queue로 구현

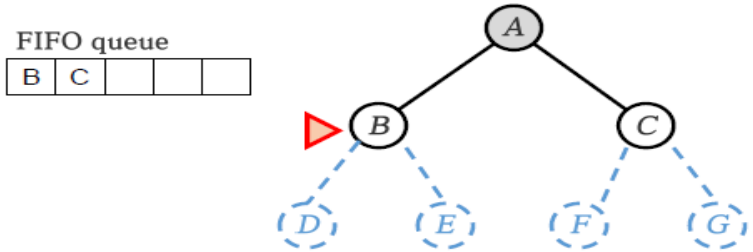




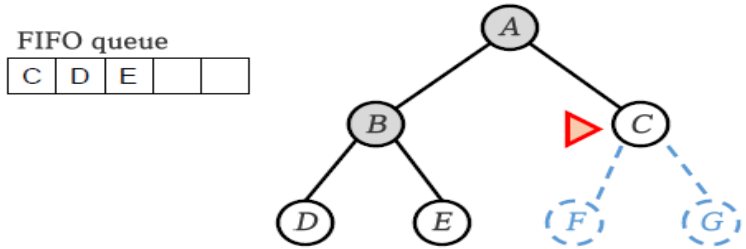
Breadth-first search(너비 우선 탐색)



1. A를 판단한다.
2. A가 goal이 아니므로 빼낸다.



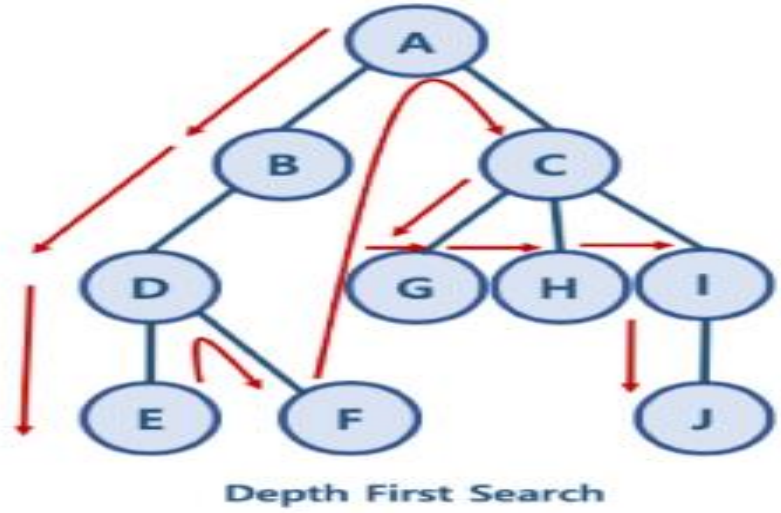
1. A에서 갈 수 있는 node를 넣는다.(B,C)
2. B가 goal이 아니므로 빼낸다.

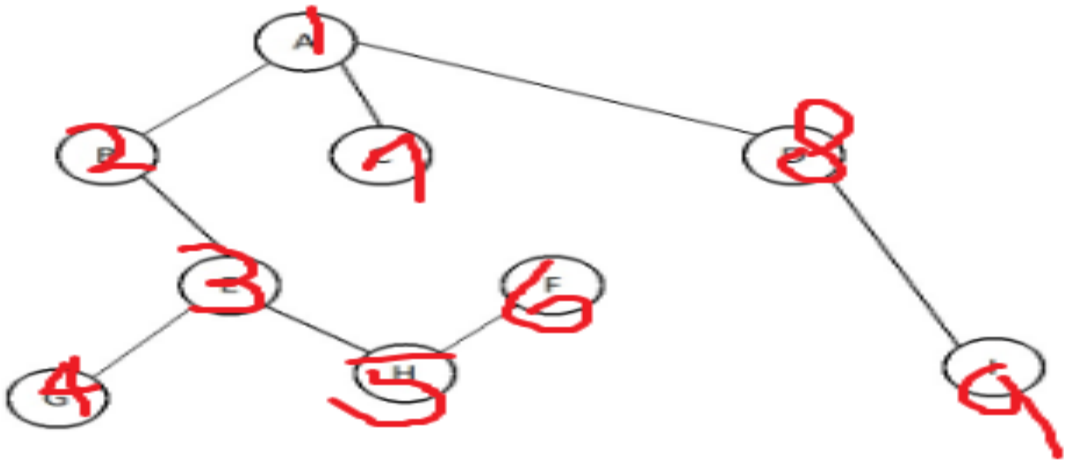


1. C를 판단한다.
2. C가 goal이 아니므로 빼낸다.
3. B가 먼저 나오므로 B에서 갈 수 있는 node(D,E)를 넣는다.

Depth-first search(깊이 우선 탐색)

- 한 노드의 자식을 타고 끝까지 순회한 후, 다시 돌아와서 다른 형제들의 자식을 타고 내려가며 순화한다.
- Deepest unexpanded node를 찾는다.
- LIFO(last in first out)





비교

Comparison

Criterion	BFS	DFS
Complete?	Yes	No
Time	$O(b^{d+1})$	$O(b^m)$
Space	$O(b^{d+1})$	$O(bm)$
Optimal?	Yes	No

- Breadth-first search is complete but expensive
- Depth-first search is cheap but incomplete