검색(searching)

검색, 정렬, 탐색

검색 : 이미 알고 있는 것을 찾는 것

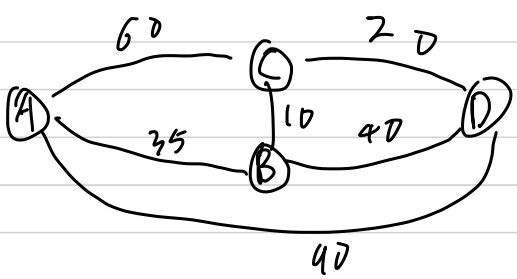
탐색 : 여러 candidates 중에서 가장 좋은(적절한) 것을 찾는 것

구조는 알고리즘과 밀접한 관계를 가진다.

검색 알고리즘, 정렬 알고리즘 : 배열구조

탐색 알고리즘 : 트리구조

탐색 : A도시에서 D도시로 가는 최단 경로?



ACD
ACBD
ABD
ABCD
AD

5의 경로를 전부 찾는다? 시간이 너무 오래 걸린다. O(n!)이다.

가능은 하지만 실제로는 불가능하다.

Possible but practically impossible

탐색 : 여러 candidate 중에서 가장 적절한 것을 찾는다.

어떤 조건이나 상황에 맞는 좋은 답이 무엇인지 찾아 나간다.

배열로 찾으면 O(n!)이다.

트리구조에서의 path가 candidates가 되는 것이고, 가장 좋은 path를 찾는 것이 목적이다.

node에서 최단 경로가 아니면 잘라버린다.

트리 탐색의 3가지 기본 방법

너비 우선 방식

깊이 우선 방식

휴리스틱 방식

너비 우선 방식:

한 레벨식 내려간다. 각레벨에서는 왼쪽에서 오른쪽으로 간다.

모든 거리가 동일하면 가장 위에 있는 것이 가장 짧은 것이다.

휴리스틱 탐색 :

인공지능 알고리즘에 속함

탐색 분야
1. tree구조
2. 다양한 tree기반의 탐색 알고리즘이 존재

문제 : 배달하는 것을 최대한 빨리하려면 어떻게 해야 하나?
해결법
1. 트리 구조로 변환하고
2. 탐색 알고리즘을 적용한다.

트리구조는 틱택토, 로봇의 작업, 바둑등에 쓰인다.
틱택토의 경우의 수 "9!"=362880
둘중 하나가 이기면 멈춘다. 그래서 255168개의 candidate가 나온다.

로봇에게 블럭을 A-B-C순서대로 쌓으라고 명령하는 경우

체스나 바둑은 복잡도가 너무커서 탐색알고리즘으로만 해결이 불가능하다.
그래서 인공지능 기법을 사용해 넣는다.

함수적 세상
output = Algorithm(input)

알고리즘은 함수를 만드는 것이다.
알고리즘이 수학적 함수보다 더 강력하다.

함수에서 다른 함수를 불러 쓸 수도 있다.

함수는 철학적이다. 그냥그렇다고 한다.

재귀적 세상 (Recursive World)
함수 내부에 자기 자신을 호출하면 계속 반복된다.
이는 유한한 도구나 시스템을 사용하여 무한한 것을 표현하고자하는 도구이다.

Factorial을 재귀함수로 표현가능하다.
```
def factorial(n):
    if n==1: return 1          #종료조건
    return n*factorial(n-1)
```

피보나치 수열, 하노이탑 문제 모두 재귀함수로 표현가능하다.