[맹목적 탐색][

깊이우선탐색 : 메모리 공간 사용 효율적, 최단 경로해 탐색 보장 불가

(스택) - 후입선출

너비우선탐색 : 최단경로해 탐색 보장, 메모리공간 사용 비효율적

(큐)-선입선출

맹목적 탐색 – (임의경로))깊이우선, 너비우선, (최적경로)균일비용

경험적 탐색 – (임의경로)언덕오르기 기법, 최상우선 탐색, (최적경로)A\*알고리즘

언덕오르기방법 : 목표노드까지의 경로비용 h(n)을 선택의 기준.

- 지역탐색 : 가장 좋은 이웃 노드 하나를 확장해 가는 탐색 방법 -> 로컬 옵티마에 빠질 수 있음.

A\*알고리즘 : 최상우선탐색 + 𝑓 (𝑛) = 𝑔 (𝑛) + ℎ (𝑛)

− 언덕 오르기 방법

• 인접한 노드 중 경로 비용 추정 값이 가장 작은 노드 선택

• Local optima에 빠질 수 있음

− 최상 우선 탐색

• 모든 확장 노드 중 경로 비용 추정 값이 가장 작은 노드 선택

• 최적 경로 보장 안됨

− A\* 알고리즘

• 시작과 현재의 비용 + 현재와 목표의 비용 추정 값이 가장 작은 노드 선택

• 최적 경로 보장

빔탐색 : 기억노드의 수를 n개 제한

Min-Max 트리 : Max -> Min -> Max 일 때, 마지막 Max는 Min이 선택한 값들 중에서 가장 큰 값을 찾는다.

{8강}

조합 최적화(Combinatorial Optimization, CO) : 어떤 목표를 달성하기 위해 최적의 매개변수 집합을 선택하는 문제를 다룸.

문제 해결 방법 : 휴리스틱 탐색 - 현재 할당의 품질(예: 위반된 제약 조건 수)을 평가하는 함수를 사용하여 탐색, 대표 기법: **Hill climbing**, **Best-first search**, **A\***, **Beam search**

로컬 탐색(Local Search) : 부분 탐색 알고리즘.

이전 해에 가까운 값을 조금씩 수정하여 해를 개선.

변수 할당을 반복적으로 개선하여 제약 조건을 모두 만족하려 함.

주로 제약 만족 문제(Constraint Satisfaction Problem, CSP)에서 **불완전한 방법**으로 사용.

**로컬 서치의 변형(Variations)**

**First Improvement** :이웃 중 **처음**으로 더 나은 해를 발견하면 바로 이동.

**Best Improvement** :모든 이웃을 살펴보고 **가장 좋은 해**로 이동.

**Random Walk** :무작위로 이웃으로 이동 (좋든 나쁘든 상관없이).

메타휴리스틱 : 로컬 최적해(local optimum)\*\*에 갇히지 않고 벗어나는 것.

더 넓은 탐색 공간(search space)\*\*을 탐험하고, 전역 최적해(global optimum)\*\*을 찾으려고 시도

**로컬 서치 기반 메타휴리스틱 (Local search metaheuristics)**

* **Simulated Annealing (SA)**
  + "냉각하는 과정"을 흉내내서, 초반에는 나쁜 해로도 이동할 수 있게 허용함.
  + 덕분에 로컬 최적에서 빠져나올 수 있어.
* **Iterated Local Search (ILS)**
  + 현재 해를 **크게 흔들어(random perturbation)** 놓고 다시 로컬 서치 시작.
* **Multi-Start Local Search (MLS)**
  + 아예 새로운 랜덤 해에서 다시 출발. 여러 번 반복해서 좋은 해를 찾음.
* **Tabu Search (TS)**
  + "탐색 금지 리스트(탐색했던 해 기록)"를 사용해서, 이미 방문했던 나쁜 해로 다시 가는 걸 막음.

**집단 기반 메타휴리스틱 (Population based metaheuristics)**

* **Genetic Algorithm (GA)**
  + 생물의 진화처럼 "선택", "교배", "돌연변이"를 반복하면서 좋은 해를 만들어냄.
* **Scatter Search**
  + 좋은 해들을 조합해서 새로운 좋은 해를 만드는 방법.
* **Memetic Algorithm**
  + GA 같은 집단 기반 방법에다가, 각 개체에 대해 **로컬 서치**까지 추가하는 **하이브리드 방식**.
* **Particle Swarm Optimization (PSO)**
  + 새 떼나 물고기 떼처럼, 여러 개체가 서로 정보를 공유하면서 최적 해를 찾는 방법.