AI알고리즘 강의 소개

수퍼스타를 키우는 곳
전주대학교

알고리즘과 모형

알고리즘의 개요

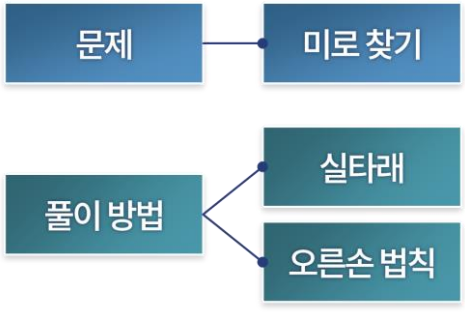
알고리즘의 개요

❖ 그리스 신화

- 젊은 청년 테세우스의 아이디어
  - 칼로 짐승을 죽이고, 실타래로 미로를 탈출

❖ 미로 알고리즘의 일반적인 방법

- 오른손 법칙
  - 오른손을 벽에 댄 상태로 진행하면 미로 탈출 가능



  - 해를 찾는 방법, 알고리즘

알고리즘의 개요

❖ 알고리즘의 유래

- 세기경 페르시아 수학자인 알 콰리즈미(Al-Khwarizmi) 이름에서 유래
  - 수학자이자 천문학자
  - 아라비아 숫자 이용해서 사칙연산과 0의 위치값 최초 사용
  - 현재까지도 선형대수학의 기초를 이루는 <알자브르 왈 무카발라> 집필

❖ 최초의 알고리즘

- BC 300년 경의 유클리드 최대공약수(GCD)
  - 두 개의 수의 공약수 중 가장 큰 수
    - 두 개의 자연수 중 큰 수에서 작은 수를 뺀 수와 작은 수와의 최대공약수를 찾는 방법

❖ 유클리드의 최대공약수 알고리즘

- 두 수의 최대공약수=큰 수에서 작은 수를 뺀 수와 작은 수와의 최대공약수

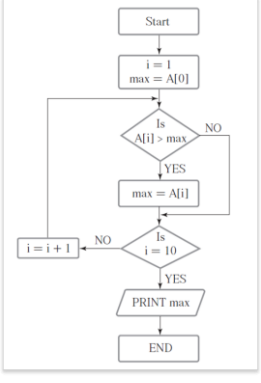
최대공약수(24, 14)
=최대공약수(24-14, 14) = 최대공약수(10, 14)
= 최대공약수(14-10, 10) = 최대공약수(4, 10)
⋮
= 최대공약수(2-2, 2) = 최대공약수(0, 2)
= 2

- 두 수를 뺀 수와 작은 수의 최대공약수를 찾는 방법
- 뺄셈 대신 나눗셈으로 최대공약수를 찾는 방법

알고리즘의 개요

❖ 알고리즘의 특성

- 정확성
  - 주어진 입력에 대해 올바른 해를 주어야 함
- 수행성
  - 알고리즘의 각 단계는 컴퓨터에서 수행 가능해야 함
- 유한성
  - 유한 시간 내에 종료되어야 함
- 효율성
  - 효율적일수록 그 가치가 높아짐
- 알고리즘 절차의 표현: 순서도(Flow Chart)



알고리즘의 개요

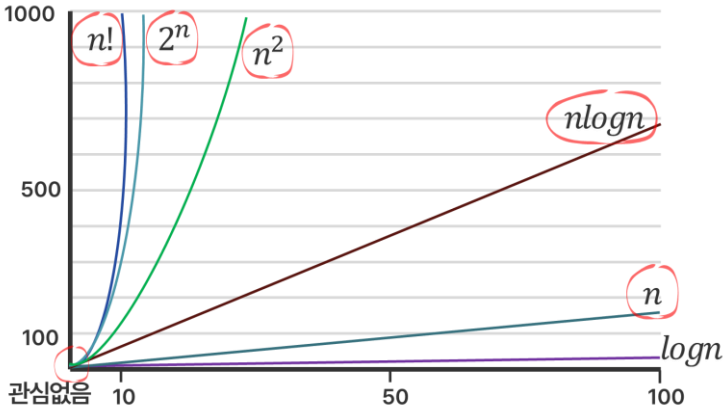
❖ 알고리즘의 효율성

- 살아남은 효율적인 알고리즘을 학습
  - 최적해를 찾는 알고리즘이 아니라면, 더 좋은 알고리즘을 찾는 형태로 발전
  - 기존의 알고리즘을 수정해서 사용함
  - 새로운 알고리즘을 만들면 인공지능의 발전이 빨라짐
- 시간과 공간을 논함
  - 시간의 소모량
    - 시간 복잡도
  - 공간의 소모량
    - 공간 복잡도
  - 알고리즘 비교 시, 시간 복잡도가 주로 사용됨
- 알고리즘 복잡도 분석에서 중요한 점
  - 알고리즘에서 입력의 크기는? (데이터의 양)
  - 복잡도에 영향을 미치는 핵심적 기본 연산은?
  - 데이터 증가로 인한 처리시간 증가는 어떤 형태인가?
  - 입력 특성에 따른 알고리즘 효율성의 차이점은?
- 알고리즘에서 입력의 크기
  - 효율성은 함수 형태로 표현
    - 예시: x의 n 거듭제곱, 다항식의 연산, Row x Col의 행렬 연산
- 실행시간의 측정 단위
  - 기본 연산
    - 예시: 더하기 연산, 대입 연산, 곱셈 연산

알고리즘의 개요

❖ 알고리즘의 효율성

- 복잡도 함수와 증가 속도



  - n의 값 증가로 알고리즘의 복잡 증가에 따라 효율성 판단
- 최선, 최악, 평균적인 효율성
  - n세 개념으로 소모 시간 분석
  - 세 개를 고려한 후 최악의 기준으로 표현하는 게 알고리즘의 수행 시간
  - 최악의 경우를 고려함

❖ 복잡도의 점근적 표기

- 시간 복잡도는 입력 크기에 대해 함수로 표기
  - O(Big-Oh)-표기
    - 복잡도 함수의 상한
  - Ω(Big-Omega)-표기
    - 복잡도 함수의 하한
  - Θ(Theta)-표기
    - 상한인 동시에 하한

알고리즘의 개요

❖ O(Big-Oh)-표기

- O-표기의 정의
  - 모든 n ≥ n0에 대해서 f(n) ≤ cg(n)이 성립하는 양의 상수 c와 n0가 존재하면, f(n) = O(g(n))
- O-표기의 의미
  - n0와 같거나 큰 모든 n에 대해서 f(n)이 cg(n)보다 크지 않다는 것

❖ Ω(Big-Omega)-표기

- Ω-표기의 정의
  - 모든 n ≥ n0에 대해서 f(n) ≥ cg(n)이 성립하는 양의 상수 c와 n0가 존재하면, f(n) = Ω(g(n))
- Ω-표기의 의미
  - n0 보다 큰 모든 n 대해서 f(n)이 cg(n)보다 작지 않다는 것

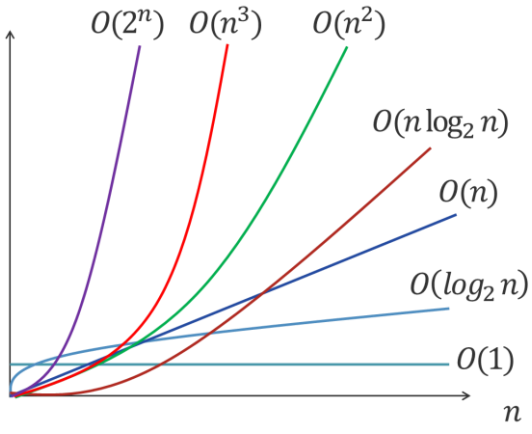
❖ Θ(Theta)-표기

- Θ-표기의 정의
  - 모든 n ≥ n0에 대해서 c1g(n) ≥ f(n) ≥ c2g(n)이 성립하는 양의 상수 c1, c2, n0가 존재하면, f(n) = Θ(g(n))
- Θ-표기의 의미
  - 수행시간의 O-표기와 Ω-표기가 동일한 경우에 사용
  - 동일한 증가율을 의미

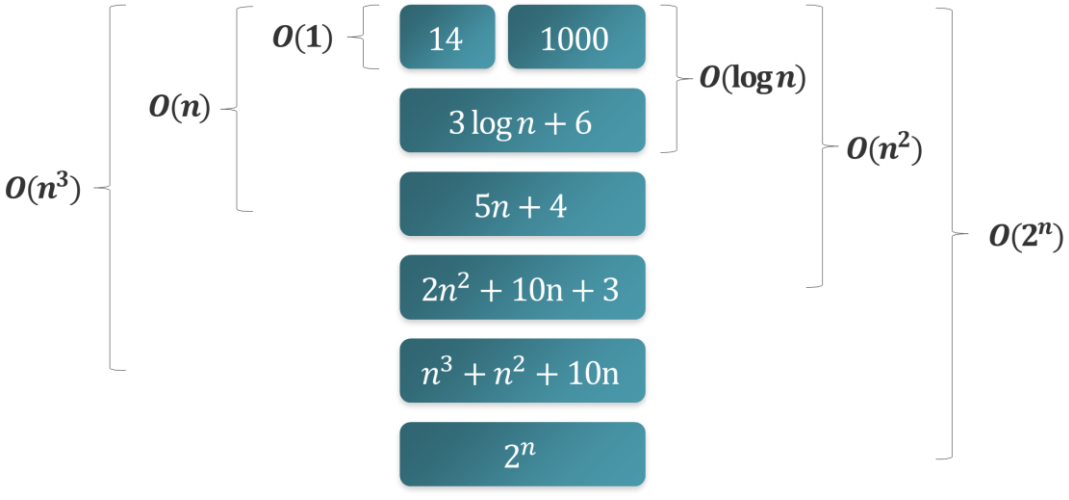
알고리즘의 개요

❖ 자주 사용하는 O-표기

- $O(1)$→상수형
- $O(\log\_2n)$→로그형
- $O(n)$→선형
- $O(n\log\_n)$→선형로그형
- $O(n\text{^}2)$→이차형
- $O(n\text{^}3)$→삼차형
- $O(2\text{^}n)$→지수형
  - n값이 커져도 직선 형태면 좋은 알고리즘



❖ O-표기의 포함 관계



알고리즘의 개요

❖ 효율적 알고리즘의 필요성

$O(n^2)$	1,000	1백만	10억
PC	<1초	2시간	300년
슈퍼컴	<1초	1초	1주일

$O(\text{nlog } n)$	1,000	1백만	10억
PC	<1초	<1초	5분
슈퍼컴	<1초	<1초	<1초

알고리즘과 모형

모형과 알고리즘 소개

모형과 알고리즘 소개

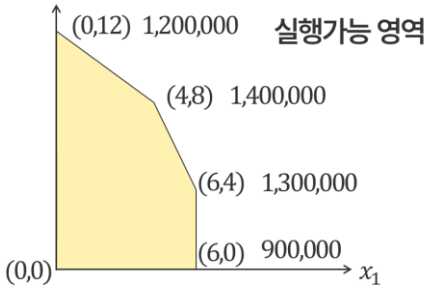
❖ 선형계획법(LP: Linear Programming)

- 자원이 제약된 상황에서 최적의 자원 활용법 탐색
- 가장 먼저 배우는 이유는 최적해를 찾는 알고리즘이기 때문
  - 도해법(Graphical Method)
  - 심플렉스법(Simplex Method)
- 형태

목적 함수와 제약식

$$
\begin{aligned}
&Max\ 150{,}000x_1 + 100{,}000x_2 \\
&s.t. \\
&\quad 2x_1 + x_2 \le 16 \\
&\quad x_1 + x_2 \le 12 \\
&\quad x_1 \le 6 \\
&\quad x_1, x_2 \ge 0
\end{aligned}
$$

도해법과 실행 가능형



- 쌍대 이론

❖ 수송모형

- 공급지에서 수요지로 제품의 물량을 최소비용으로 이동시킬 수 있는 최적 대안선택 문제
- 근사해를 찾는 알고리즘 제작법 학습
  - 알고리즘 변형 및 아이디어 선택의 통찰력 향상
  - 북서코너법, 보겔법, 러셀법, MODI 법 등

모형과 알고리즘 소개

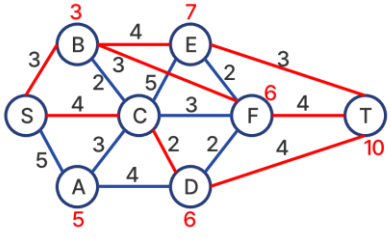
❖ 정수계획법(Integer Programming)

- 선형계획법에 정수 조건이 포함
  - 분단탐색법(Branch and Bound Method)과 선형계획법이 혼합된 방법
    - 최적해를 찾는 방법과 혼합

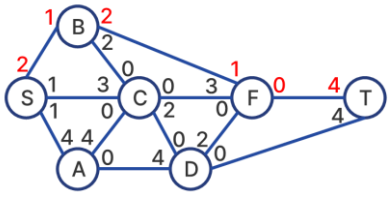
❖ 네트워크모형(Network Model)

- 최소 걸침나무 문제(Spanning Tree Problem)
  - 효율적으로 최적해를 찾는 법

최단경로문제(Shortest Path Problem)

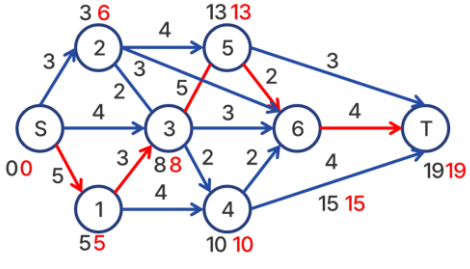


최대흐름문제(Maximal Flow Problem)



- 대형 프로젝트 진행 시, 선후 관계를 고려한 공사 기간의 예측

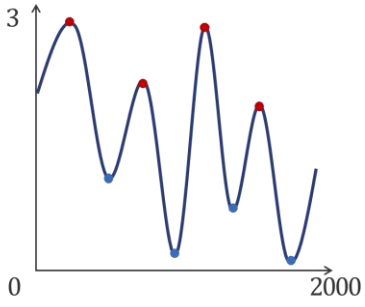
PERT/CPM(Program Evaluation and Review Technique/Critical Path Method)



모형과 알고리즘 소개

❖ 비선형계획 문제(Nonlinear Programming Problem)

- 선형이 아닌 비선형으로 표현된 최적화문제



$$Minimize \ \ z(x) = x_1^2 + x_2^2 + x_3^2$$
$$s.t \quad x_1 + x_2 + x_3 = 6$$
$$Max\, x_1 x_2$$
$$s.t. \quad 2x_1 + x_2 \le 16$$
$$x_1, x_2 \ge 0$$

- 미분
- Hessian 행렬식
- 라그랑지 방법
- 쿤-터커 조건법

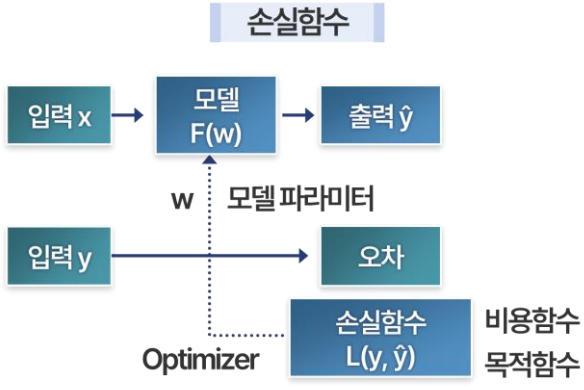
❖ 발견적 해법(Heuristic Method)

- 근사해 개념에서 발전된 개념
  - 최적해를 찾지 못하여, 반복적인 알고리즘으로 최적해의 근사값을 찾음
    - 군집 탐색
    - 의사결정나무
    - 게임이론
    - 메타휴리스틱 기법

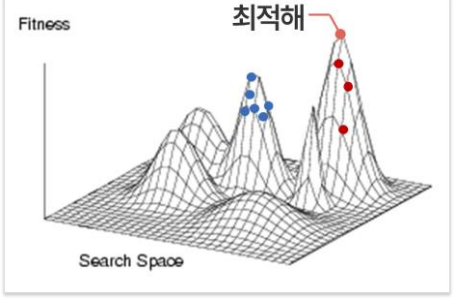
모형과 알고리즘 소개

❖ 손실함수와 경사하강법

- 손실함수를 줄여 근사해 도출하는 경사하강법



- 학습률
  - 실제 예제를 통한 학습률의 의미 파악 및 변화에 대한 학습
- 유전자 알고리즘(Genetic Algorithm)
  - 유전자를 통해 발생하는 일을 최적화 문제에 적용

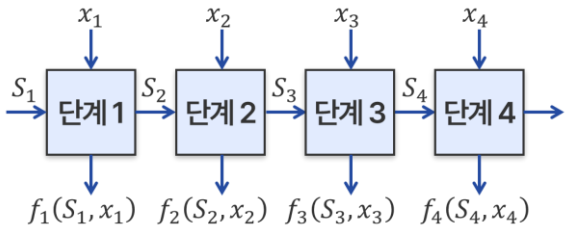


- 유전 개념의 도입으로 최적해를 도출

모형과 알고리즘 소개

❖ 동적계획법(Dynamic Programming)

- 전체 문제의 부분화로 해를 도출



$$f_n(i) = \text{Max}\{v_n x_n + f_{n+1}(i - w_n x_n)\}$$

- 배낭문제
- 와그너-위튼(Wagner-Whitin: W-W) 모델

❖ 마아코프 분석(Markov Analysis)

- 현재를 기반으로 미래를 예측
  - 마아코프 체인
  - 관찰 가능한 것에서 유추하는 방법론을 제시한 은닉 마아코프 모델