알고리즘[1]

- 실전적 문제해결 중심으로 -

Algorithms

for

Practical Problem Solving

부산대학교 정보의생명대학
정보컴퓨터 공학부
조환규 교수[2]
hgcho@pusan.ac.kr

---

1) version 1(d), finally updated 2025. Dec. 29.
2) 강사의 신분으로 이 알고리즘 교재에서 사용된 sample code {c++, python}가 필요한 경우에는 간단한 신분 증명과 함께 email, 전화번호를 저자(hgcho@pusan.ac.kr)에게 요청하시면 받으실 수 있습니다.

차례

서문

알고리즘, 좀 더 정확하게 말해서 컴퓨터 알고리즘(computer algorithm)은 컴퓨터 과학(Computer Science)의 가장 핵심적인 과목이며, 다른 관련 전공과의 차이를 구분해주는 signature 교과목이다.[3] 따라서 컴퓨터 과학, 컴퓨터 공학 전공자들은 어떤 교과목보다 이 컴퓨터 알고리즘 교과의 그 본질적인 내용에 대하여 익혀야 한다. 동시에 실제적인 프로그래밍을 통하여 알고리즘의 효율성을 이해하는 것인 현실적으로 큰 도움이 된다는 사실을 경험해야 한다.

초기 컴퓨터 알고리즘은 알고리즘 분석 위주, 즉 수학 기반의 모형으로 교육되어 왔다. 예를 들어 알고리즘 복잡도 분석을 위한 다양한 asymptotic analysis 기법이 교육되었다. 이후 1990년대 넘어 컴퓨터가 보편화되고 PC가 보급되면서 보다 실용적인 관점으로 알고리즘이 교육되었다. 그 대표적인 분야가 단순히 이론적인 분석이 아니라 실제 machine에서 수행시간과 사용한 resource의 양을 측정하고 비교하는 실험적 알고리즘학(experimental algorithmics)이다. 이 연구가 확장되어 이후 알고리즘 공학론(algorithm engineering)이라는 새로운 학문이 탄생하게 된다. 알고리즘 공학론은 다양한 기계에서 다양한 언어로 구체적인 알고리즘을 구현하였을 때 실제로 나타나는 성능과 여러 현상을 체계적으로 분석하는 학문으로 2000년 이후에 매우 중요한 분야로 다루어지고 있으며 관련하여 국제적인 학술대회도 열리고 있다.

전문 연구자가 아닌 일반 현장 개발자에게 요구되는 “알고리즘 분석/개발 능력”은 어떤 것일지 생각해보자. 대부분 학사 학위 수준의 현장 연구자들에게 필요한 알고리즘 분석 능력을 당면한 문제가 intractable 한 NP-hard 류의 문제인지를 여부를 판단하는 것이다. AI, 기계학습에서 사용되는 multi-variable optimization 문제 대부분은 아직 Polynomial time 알고리즘이 개발되지 못한 NP-complete 문제가 대부분이다. 이런 문제의 특성으로 올바르게 이해하지 못하고 대강의 heuristics로 문제를 해결할 경우 궁극에서 심각한 상황에 이를 수 있다. 일단 연구자들은 자신의 문제 NP-complete 류의 문제가 아닌지 증명할 필요까지는 없지만, 그 문제의 골격이 이미 알려진 NP-complete 류의 문제와 유사함은 인식하고 이를 인터넷의 resource를 통하여 확인할 정도는 되어야 한다.

---

3) 물리학과에서 양자역학, 전기/전자공학에서의 전자기역학, 분자생물학과의 분자유전학과 같은 위치를 컴퓨터 알고리즘이 차지하고 있다.

현장 연구자의 또 다른 능력은 개발한 시스템 성능의 예측이다. 이 과정은 time complexity 분석의 기초이다. 예를 들어 데이터의 크기가 2배씩 늘어날 때 시간이나 필요한 공간은 얼마나 더 늘어날 것인지에 대해 예상을 하고, 그 예상이 실제 현실에서 그대로 나타나는지를 확인하는 것은 매우 중요한 일이다. 예상보다 너무 빨리 작업이 종료되어도, 너무 늦게 종료되어도 모두 문제이다. 이것은 단순히 성능의 문제가 아니라 그 알고리즘의 정확성에 심각한 문제가 있다는 것은 암시하는 것이다. 코드를 실제 수행해봐야만 그 결과를 알 수 있다는 식의 태도는 알고리즘의 핵심에 대한 이해가 없는 사람들이 가지는 전형적인 태도이다. 본 강의를 통하여 모든 수강생은 자신의 코드가 어떻게 동작하고 얼마나 효율적으로 동작하는지를 예상할 수 있는 다양한 기법에 대하여 배울 것이다.

컴퓨터 알고리즘에 관한 교재 형식은 크게 3가지 유형이나. 가장 전통적인 형식은 알고리즘 개발 방법론, 예를 들어 분할정복법, 욕심쟁이법 (Greedy Method), 동적계획법 Dynamic Programming, 완전탐색법(Exhaustive searching), Branch and Bounding, General Heuristics 등의 대표적인 알고리즘 설계 방법론(methodology)을 순차적으로 강의하는 것이다.

알고리즘 교재의 또 다른 한 유형은 알고리즘 개발 방법론 강의 대신 특정 문제, 데이터 중심으로 관련된 알고리즘을 교육하는 것이다. 예를 들어 수치 알고리즘, 그래프 알고리즘, 정수론 알고리즘, 병렬 알고리즘 순서로 편성된 교재도 있다. 앞의 개발 방법론 기반의 알고리즘 교재에 비해서 좀 더 현실적이고 학부생 수준에 더 적합하지만, 기본적인 알고리즘 모형이나 원리적인 부분이 부족한 것이 사실이다. 마지막 또 다른 형식은 2번째 유형을 좀 더 확장하여 대표적인 독립적 알고리즘 50여 개를 선택하여 하나하나씩 설명하는 것이다. 예를 들어 Euclid 알고리즘 등 역사적으로 유명한 계산법을 알고리즘 모형으로 각각 설명한다. 각 장으로 앞의 장과 독립적으로 구성되어 있어서 순차적으로 보지 않아도 되며 흥미로운 장부터 선택적으로 살펴봐도 별문제가 없다.

이 3번째 방식, 주요 토픽별 알고리즘 설명은 교육용 교재라기보다 독습 교재에 더 적합하지만 수학 모형에 익숙지 않은 일반 연구자, 수리과학과 거리가 있는 사람들에게는 매력적인 전개 방식이다. 이번 개발하는 교재는 가장 전통적인 교재 형식에 기반하고 있지만 3번째 형식도 일부 적용하여 계산기하학(Computational Geometry), 확률기반(Randomized) 알고리즘을 추가하였다.[4)]

4) 계산기하학은 이전부터 추가된 내용이고 2022년부터 추가된 것은 확률 알고리즘이며 앞으로 여유가 있을 경우 병렬 알고리즘(parellel algorithm)도 추가할 예정이다.

알고리즘의 중요한 부분이지만 본 교재에 포함되지 못한 알고리즘은 다음과 같다. Neural Network Algorithm, 수치 알고리즘(Numerical Algorithm), Algebraic Algorithm(정수론 기반의 알고리즘), Online 알고리즘 등은 포함되지 않았다. 물론 학부 수준에서 이 모두를 설명하는 교재는 없지만 추후 포함된 위 알고리즘은 "알고리즘 연구의 최신 동향 (Recent Topics on Algorithm Researches)"으로 독립된 장에 간략하게 설명할 예정이다.

이번 부산대학교에서 제공되는 알고리즘 강의는 바로 위에서 설명한 실험적 알고리즘학의 목표와 같은 목표로 준비되고 강의 된다. 따라서 현장에서 나타나는 다양한 문제를 알고리즘으로 구현하고 그 성능을 보편적인 컴퓨터, 즉 windows 기반의 PC와 LINUX 기반의 NESPA 서버에서 평가하고자 한다. 또한 많은 학생이 관심을 가지는 입사 코딩 수준의 알고리즘 능력 평가를 대비하기 위하여 면접용 알고리즘 평가 문제를 다수 다루어볼 예정이다.[5] 또한 알고리즘 능력을 평가하는 다양한 기출 문제, 이전 학기에 제공된 알고리즘 코딩 과제물을 본 교재의 부록에 모두 포함했다.[6]

모쪼록 이 과목을 수강한 모든 학생은 이번 교재를 통하여 실용 기반의 알고리즘 분석의 기법을 이해하고 배운 지식을 실전에서 활용할 수 있는 전문가가 되기를 기원해 본다.

5) 요즘 입사 코딩의 기본은 대부분 자료구조, 알고리즘을 중심으로 평가되고 있어 이 수준에 맞추어 준비하는 것은 현실적인 도움이 된다. 즉 time complexity hierarchy(위계) 등은 알고리즘 연구에서 매우 중요한 주제이기는 하지만 대부분의 입사 면접에서 이런 내용은 다루어지지 않는다.

6) 2018년 이후에 사용된 알고리즘 코딩 과제물이 부록에 제시되어 있으며 본 교재 강사인 조환규 교수가 20여 년간 국제대학생 프로그래밍 챌린지(ICPC)에서 출제한 예선 문제, 본선 문제를 해설과 함께 부록에서 제공한다.

수강생을 위한 프로그래밍 과제물 표절 기준[7]

a. 과제물이 일정 수준 이상으로 유사한 이유에 대하여 수강생이 상식적 수준으로 해명하지 못하면 그에 관계된 모든 학생은 처벌받습니다. 설사 다른 친구에게 단순 참고하게 하려는 의도로 소스 코드를 전해준 경우에도 그 의도와 상관없이 처벌받습니다. 따라서 모든 수강생은 이런 가능성이 없도록 해야 합니다.

b. 과제에 따라서 그 완성을 위하여 여러 사람이 의논하는 것은 권장하지만 이 경우 최종 결과물이 유사할 가능성에 대해서는 반드시 수강생 각자가 제출 전에 확인해야 합니다.

c. 비록 두 사람이 공모하지 않고 외부에 공개된 자료(공개 소스)를 참고하면 결과물의 소스 코드는 유사할 수 있습니다. 이와 같은 "간접 표절"을 방지하기 위하여 외부에서 받는 코드는 **충분히 자신의 관점으로 수정하여 간접 표절이 발생하지 않도록** 특별히 유의해야 합니다. 이러한 간접 표절도 부정행위의 한 경우로 처벌받을 수 있습니다. 만일 외부 코드를 활용한 경우에는 그 출처와 자신이 개선한 사항을 반드시 설명해야 합니다.

d. 소스 코드 표절(의도된 경우든 아니든)이 확인된 경우에는 관련된 모든 수강생의 과제물은 0점 처리되며 다시 반복될 때 과목성적은 이유 여하를 불문하고 F로 처리됩니다.

e. 중간 및 기말고사에서의 부정행위, 예를 들어 대리시험, 시험 중 폰이나 인터넷 사용 등은 교칙에 따라서 엄중하게 처벌을 받으므로 절대 이런 일에는 조금의 관심도 가지지 않기를 바랍니다.

f. 본 교육의 NESPA[8] 강의 사이트는 학생들의 선의에 기초하여 구성되어 있으므로 높은 수준의 해킹 방지 모듈이 설치되어 있지 않습니다. 따라서 강의 관련 사이트를 해킹하거나 해킹을 시도(trial)하는 경우에도 교칙에 따라서 엄중한 처벌을 받게 됩니다. 또는 시도한, 시도하려는 NESPA 해킹 방법을 다른 친구에게 알려주는 경우도 처벌의 대상이 됨을 인지하기 바랍니다.

---

7) 이 교재는 2023년 봄학기 강의를 기준으로 작성되었다.
8) https://topaz.pusan.ac.kr/algo2025/ 이전 ESPA는 2021년 가을학기 기준으로 NESPA로 새롭게 구성되었다.