0. Introduction

0.1 The Role of Algorithms

Terminology

- Algorithm: 특정 작업을 수행하기 위해 정의한 과정의 집합
- Program: Algorithm을 표현한 것
- Programming: Program을 개발하는 과정
- Software: Program과 Algorithm
- Hardware: 장비 또는 기계

History of Algorithms

- 알고리즘에 대한 연구는 원래 수학에서 비롯됨
  - Long division algorithm(세로 나눗셈)
  - Euclidean algorithm(최대공약수 구하기)
    1. M과 N에 각각 큰 수, 작은 수를 넣는다.
    2. M을 N으로 나눈 나머지를 R에 넣는다.
    3. R이 0이 아니라면 M에 N을 넣고 N에 R을 넣고 step 2로 돌아가고, R이 0이면 구하고자 하는 최대공약수는 N이 된다.
- Gödel's Incompleteness Theorem: "어떤 문제들은 알고리즘을 통해 해결할 수 없다."

0.2 The History of Computing

Early computing devices

- Abacus(주판): 각 구슬이 숫자를 의미
- Gear-based machines(1600s~1800s): 각 기어가 숫자를 의미

Early data storage

- Punched cards: 카드에 구멍을 뚫고, 구멍의 위치가 데이터를 의미
  - 1801년에 Jacquard Loom에 의해 처음 사용. 천을 짤 때의 패턴을 저장하기 위해
  - Babbage's Analytical Engine에서 프로그램의 저장소 역할
- Gear positions: 기어의 위치가 데이터를 의미?

Early computers

- Mechanical relay 기반
- Vacuum tube 기반
  - ENIAC

Personal Computers

- 1981년에 IBM에 의해 등장
  - 현재 데스크톱 컴퓨터의 표준 디자인
  - 대부분 Microsoft의 소프트웨어 사용

Into the Millennium

- 인터넷이 통신을 진화시킴
  - WWW, 검색 엔진 등
- 컴퓨터의 소형화
  - 임베디드(차량의 GPS), 스마트폰

0.3 An Outline of Our Study

1. Data Storage
2. Data Manipulation
3. Operating Systems
4. Networks and the Internet
5. Algorithms
6. Programming Languages
7. Software Engineering
8. Data Abstractions
9. Database Systems

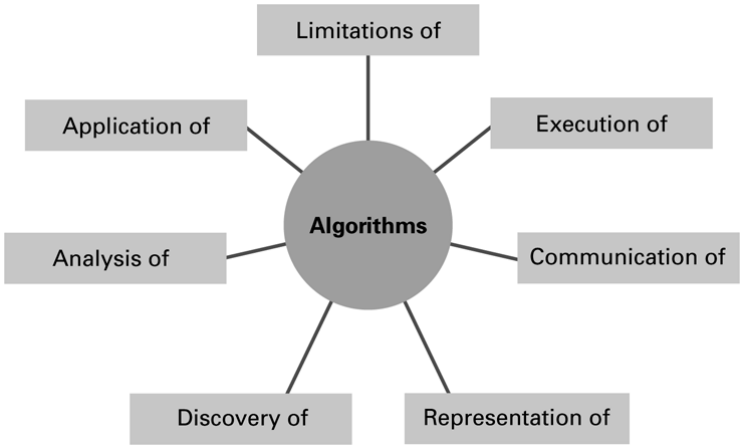
10. Computer Graphics
11. Artificial Intelligence
12. Theory of Computation

0.4 The Overarching Themes of Computer Science

- Computing technology는 governments, economics, scientific research, role of data, communication 등 여러 분야에 영향을 미침
- Computer science를 통합할 수 있는 7가지 "Big Idea": **Algorithms, Abstraction, Creativity, Data, Programming, Internet, Impact**

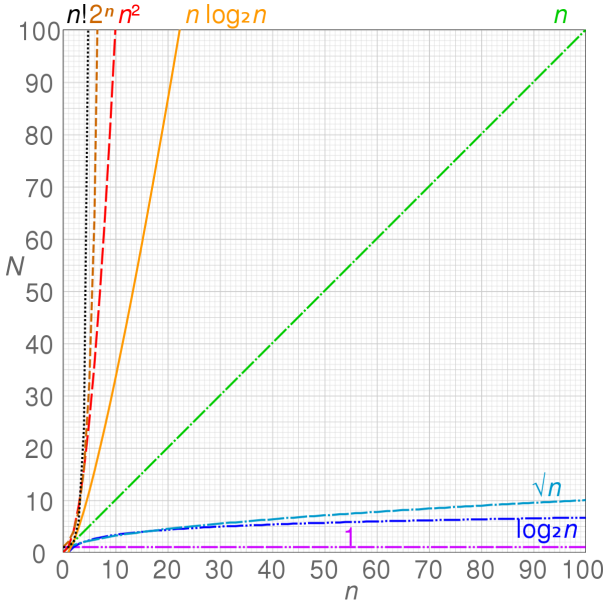
Algorithms

- 특정 작업을 수행하기 위해 정의한 과정의 집합
- 다른 과목들에서 활용됨(Ex. Mathematics, Engineering, Psychology, Business Administration)
- Computer science에서 알고리즘의 중심적 역할



  - 어떤 문제들이 알고리즘 절차를 통해 해결될 수 있는가?
  - 어떻게 하면 알고리즘에 대한 발견을 더 쉽게 할 수 있는가?
  - 알고리즘에 대한 표현 및 소통 방식을 어떻게 더 향상시킬 수 있을까?
  - 서로 다른 알고리즘에 대한 특성을 어떻게 분석하고 비교할 수 있을까?
  - 정보 조작을 하는데 있어 알고리즘이 어떻게 사용될 수 있을까?
  - Intelligent behavior(AI?)를 만들기 위해 알고리즘을 어떻게 적용할 수 있을까?
  - 알고리즘의 적용이 사회에 어떻게 영향을 미칠까?

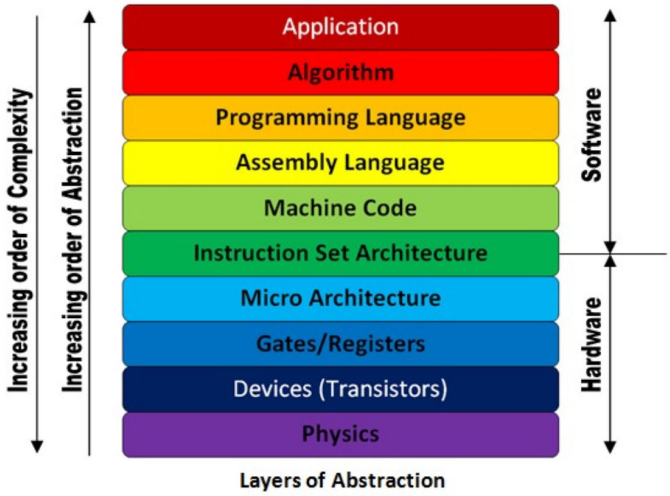
- 알고리즘 분석
  - 시간 복잡도



Abstraction

문제를 어떻게 표현하느냐(추상화)

- Abstraction: ???
- Abstract tool: ???
- Abstraction의 단계
  -



Creativity

- 컴퓨터 과학은 창의적인 과목

    - 알고리즘의 발견 및 적용은 사람에 의해 행해지므로 다양한 방법으로 표현 가능
- 큰 소프트웨어 시스템을 개발하는 것은 거대한 조각상을 구상하는 것과 같음

Data

- 컴퓨터는 어떤 정보든 표현할 수 있음(이산화 및 디지털화)
- 알고리즘은 데이터를 처리하고 변환함
- 대용량 저장소와 고속 네트워크 필요
- Questions
    - 컴퓨터는 숫자, 문자, 이미지, 소리, 영상과 같은 일반적 디지털 자료를 어떻게 저장하는가?
    - 컴퓨터는 실제 세계의 아날로그 정보를 어떻게 근사화하는가?
    - 컴퓨터는 에러를 어떻게 발견하고 방지하는가?
    - 처리하는 디지털 데이터의 양이 무한히 커져가는데, 이에 대한 결과는 어떻게 될까?

Programming

- Programming = 사람의 의도를 실행 가능한 알고리즘으로 번역하는 것
- 컴퓨터 하드웨어는 간단한 알고리즘 단계만 가능
- 프로그래밍 언어에서의 Abstraction은 사람이 복잡한 문제를 해결하는 데 도움을 준다.
- Questions
    - 프로그램은 어떻게 빌드되는가?
    - 어떤 오류들이 프로그램에서 발생할 수 있는가?
    - 프로그램에서의 에러는 어떻게 발견되고 고칠 수 있는가?
    - 현대 프로그램에서 에러가 미치는 영향은 무엇인가?
    - 프로그램을 어떻게 문서화하고 평가하는가?

Internet

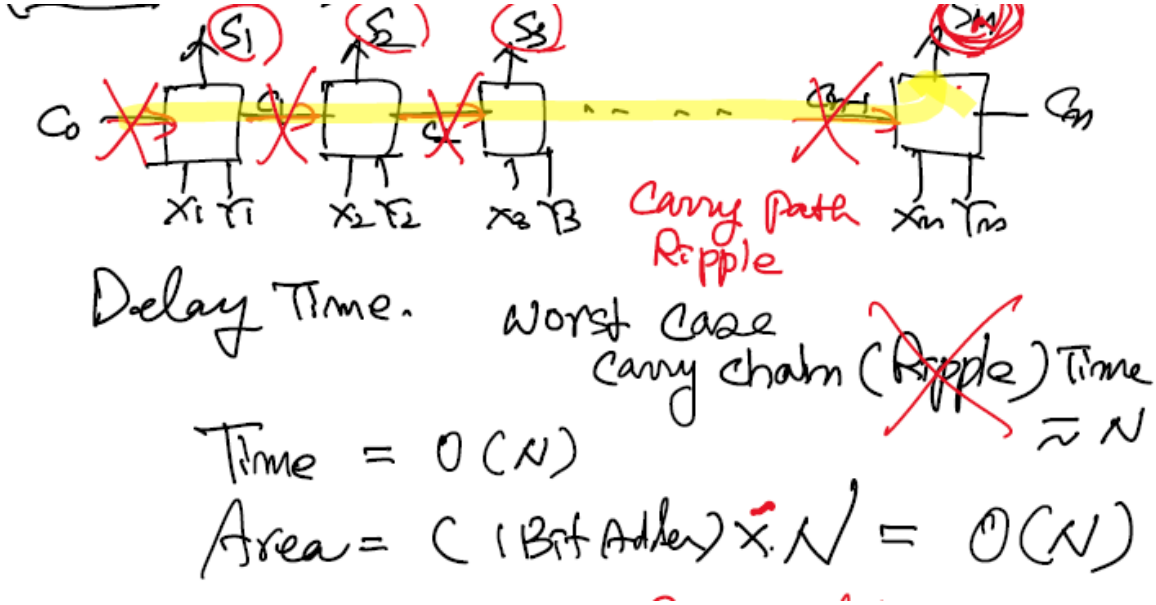
- 정보의 저장, 검색, 공유에 깊은 영향을 끼침
- 사생활 및 보안 문제

Impact

- 사회적, 도덕적, 법적으로 미치는 영향
  - 보안 이슈
  - 소프트웨어 소유 및 저작권
  - DB 기술이 미치는 영향
  - AI 기술의 진화에 따른 결과
  - "정답"은 없지만, 다음에 대한 주의가 필요하다.
    - 다양한 이해관계
    - 대안
    - 단기적 및 장기적 결과
  - Character-based ethics: "Good Behavior" is a consequence of "Good Character"

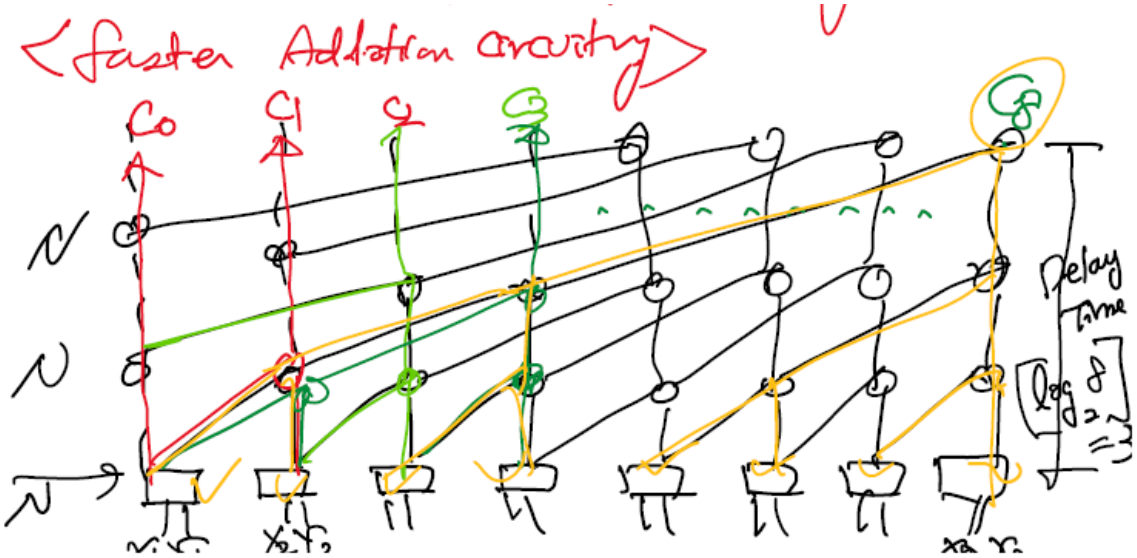
번외

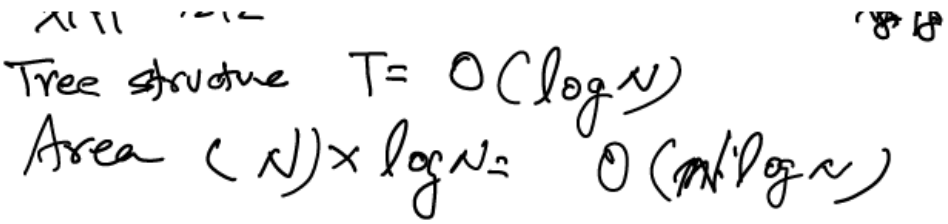
Ripple Carry Adder



Tree Adder

- RCA보다 더 빠르게 작동하는 구조, 그러나 공간은 더 많이 차지





	Area Complexity	Time Complexity
Ripple Carry Adder	O(N)	O(N)
Tree Adder	O(NlogN)	O(logN)