§1 Recap, Intro & Overview

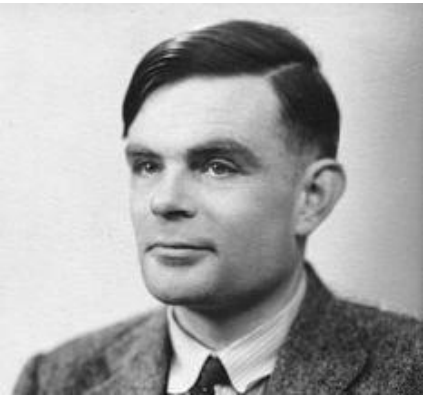
- Philosophy of (Computer) *Science*
- Power of Abstraction, Hardware vs. Math
- (Importance of) Asymptotic efficiency
- Sorting Example: specification & optimality
- Algorithm Analysis: How and Why?
- Course Overview and Administration
- Pedagogy & Teaching Philosophy

§1: CS ↔ EE

1936: Alan Turing introduces "Turing Machine"

1941: Konrad Zuse's Z3 becomes operational

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM



a) Define mathematical model of "digital computer"

b) Demonstrate fundamental capabilities ("*universal TM*")

c) Prove ultimate limitations ("*Halting Problem*")

d) Guide to Engineer./Implement.

§1: CS ↔ EE

1936: Alan Turing introduces "Turing Machine"

1941: Konrad Zuse's Z3 becomes operational

1931: First practical general-purpose analog computer operational at MIT (18 indep.variables)

1941: Claude Shannon publishes "*Mathematical Theory of the Differential Analyzer*"

1994: Peter Shor develops a "*quantum* algorithm" for factoring integers in polynomial time

2004ff: *Goldrush* realizing quantum computers

§1 "Virtues" of Computer Science

distinguishing from
Electrical **Engineering**

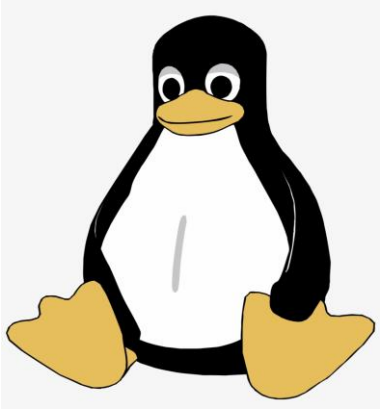
≠program, ≠heuristic

- problem specification
- algorithm design
  (primitives, semantics, cost)
- and analysis
  (correctness, efficiency)
- optimality proof (cmp. $\mathcal{P}/\mathcal{NP}$)

§1 Power of Abstraction

high-level program.

obj.library

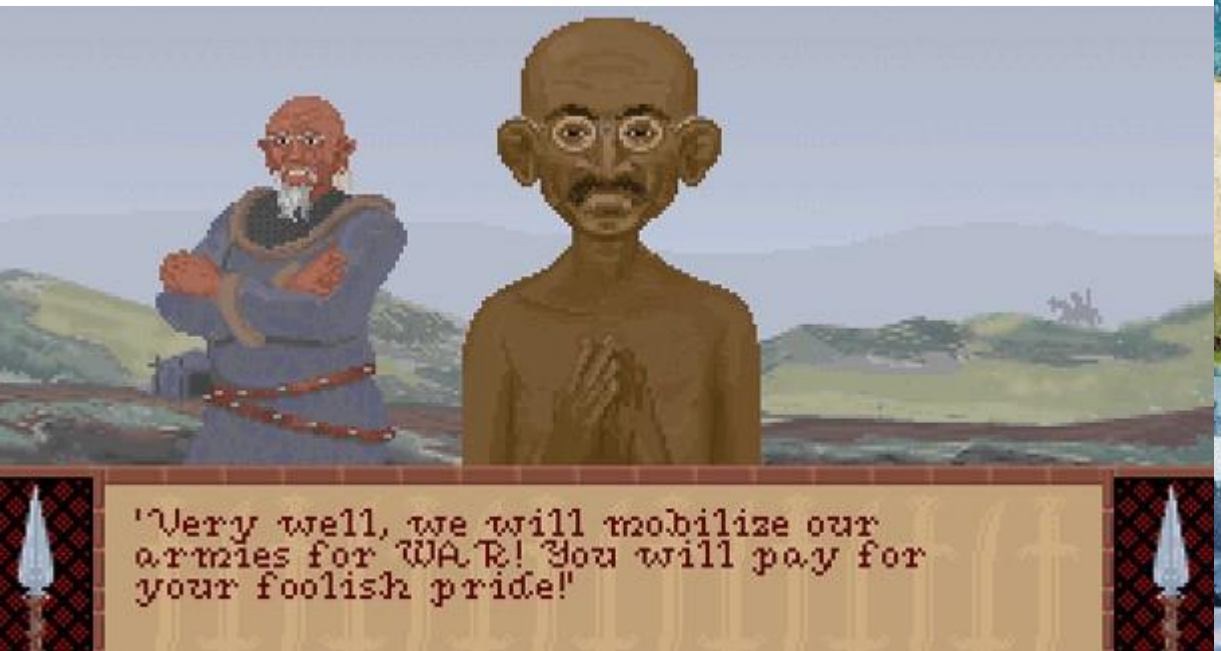


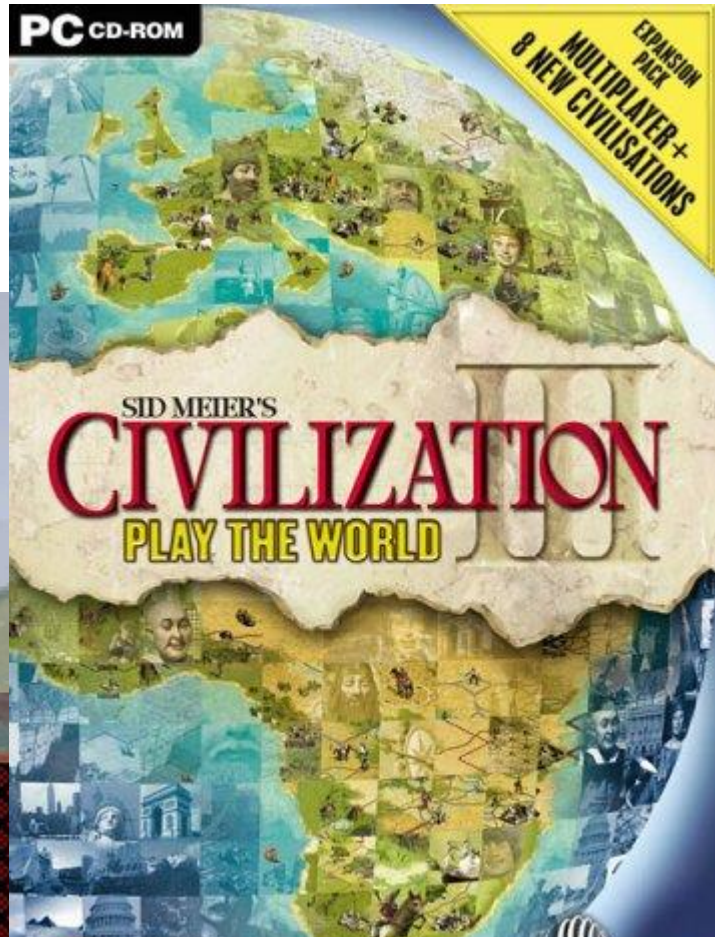
Computer Hardware

§1 Hardware vs. Math Data Type

- Each country/leader described by "A.I." character parameters.
- Initially *Gandhi*.**aggression** := 1
- When country adopts democracy, **aggression -= 2**.

(signed) byte/word $\neq$ ($\mathbb{Z}$) $\mathbb{N}$





§1 Recap

asymptotic efficiency

n	$\log_2 n \cdot 10$s	$n \cdot \log n$ sec	n^2 msec	n^3 µsec	2^n nsec
10	33sec	33sec	0.1sec	1msec	1msec
100	≈1min	11min	10sec	1sec	40 bill.Ys
1000	≈1.5min	≈3h	17min	17min	
10 000	≈2min	1.5 days	≈1 day	11 days	
100 000	≈2.5min	19 days	4 months	32 years	

Algorithm *analysis*,

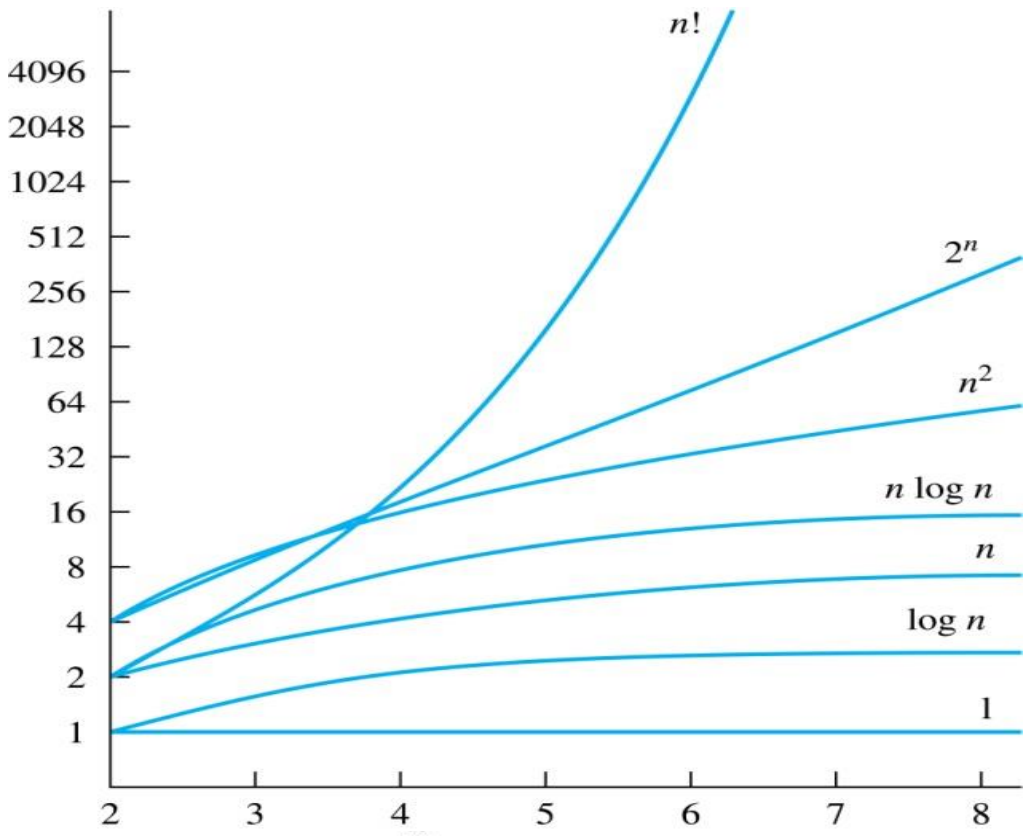
as opposed to empirical program evaluation

„Premature optimization is the root of all evil"
(D.Knuth)

§1 Asymptotic Growth

Landau „big-Oh“ notation $f = O(g)$

- constant 17
- logarithmic $\log(n)$
- square root $\sqrt{n}$
- linear n
- quasilinear $n \cdot \log(n)$
- quadratic n^2
- cubic n^3
- polynomial $c \cdot n^c$
- superpolynomial $n^{\log(n)}$
- exponential 2^n
- doubly exponential 2^{2^n}
- tetration $2^{2^{\cdots^{2^n}}}$



§1 Specification Example: Sorting

Specification: Fix set X <u>with total order</u> $\leq$.

Input: $N \in \mathbb{N}$ and array $x[1 \ldots N]$ with values in X

Output: *Permutation* $\pi : [1 \ldots N] \to [1 \ldots N]$ s.t. $x_{\pi(1)} \leq \ldots \leq x_{\pi(N)}$

§1 Optimality Example: Sorting

Specification: Fix set X with total order $\leq$.

Input: $N \in \mathbb{N}$ and array $x[1 \ldots N]$ with values in X

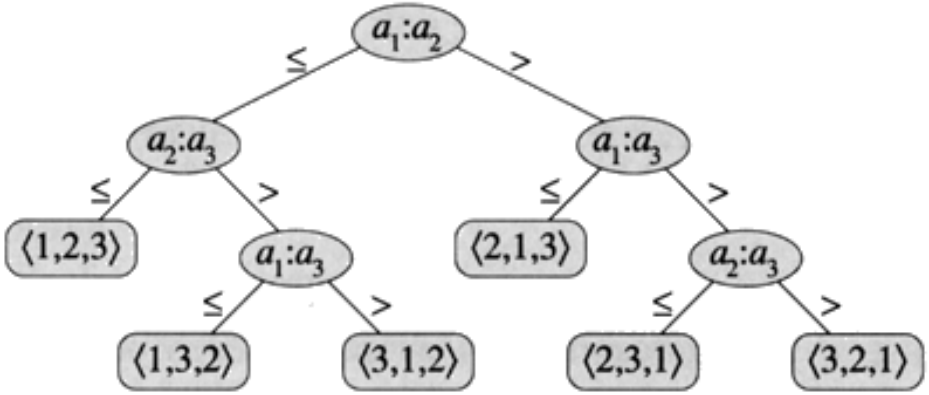
Output: *Permutation* $\pi:[1 \ldots N] \to [1 \ldots N]$ s.t. $x_{\pi(1)} \leq \ldots \leq x_{\pi(N)}$

Lemma: a) For fixed N, every *comparison-based* sorting algorithm of worst-case runtime $T(N)$ can be "unrolled" into a *decision tree* for sorting N elements of depth $\leq T(N)$.

b) Every permutation $x=\pi$ ends up in the unique leaf with label π^{-1}.

c) A binary tree with $N!$ leaves has depth $\geq \log(N!) = \Theta(N \cdot \log N)$

Definition: A *Decision Tree for sorting N elements* is a binary tree whose internal nodes are labeled with tests "$x[i] \leq x[j]$?" and whose leaves are labeled with permutations π such that every input x "ends up" in a leaf whose label π satisfies $x[\pi[1]] \leq \ldots \leq x[\pi[N]]$.



§1 Course Goals

Design and <u>Analysis</u> of Algorithms

with respect to various <u>notions</u> of <u>performance</u>:

worst-case,

average-case,

expected-case,

amortized;

competitive ratio,

approximation ratio etc.

(sequential) runtime

memory

parallel runtime

communication volume

#processors

…

§1 *Purpose* of Algorithm Analysis

Tradeoffs: *polynomial* vs. $O(\ldots)$ vs. constant factors;
Count #operations, bit-cost, w/o access latency etc.

Goal: Predict the behavior of an algorithm

- before actual execution
- independent of hardware details (e.g. clockrate)
- realistically but simple.
- Indicate im/possibility of improvement/optimality.

Cost Measures:

(sequential) time

parallel runtime

memory use

#processors/gates

(communication volume)

etc …

§1 Course Overview

Design and Analysis of Algorithms

1. Recap, Intro & Overview
2. Tree Data Structures
3. Average-Case Analysis
4. Amortized Analysis
5. Randomization/Expected Case
6. Online/Competitive Analysis
7. Complexity Theory Intermission
8. Approximation
9. Parallel Time
10. Memory

§1 Course Administration

Course Homepage

http://kaist.theoryofcomputation.asia/24cs500

ELICE programming assignments

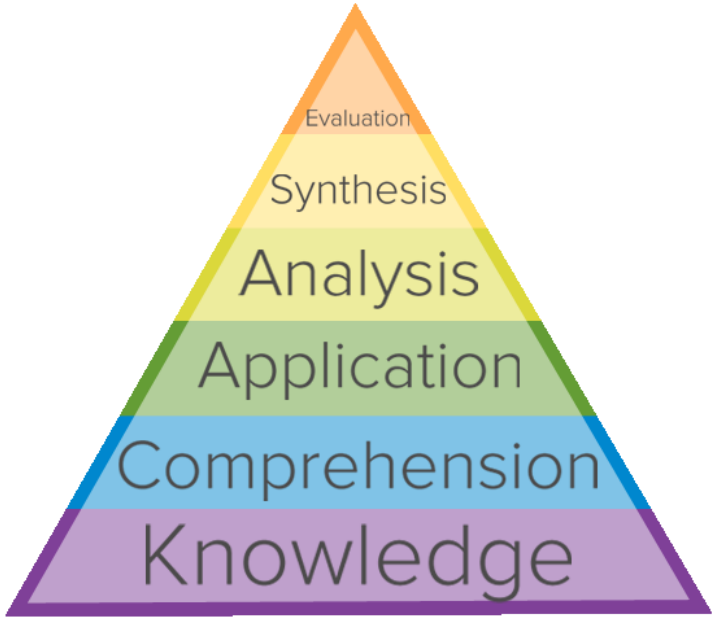
http://kaist.elice.io/

KAIST Learning Management KLMS

http://klms.kaist.ac.kr/

§1 Pedagogy

Bloom's Hierarchy
of cognitive learning:



Konrad Lorenz:
(Nobel Prize 1973)

- *What is thought is not said*
- *What is said is not heard*
- *What is heard is not understood*
- *What is understood is not believed*
- *What is believed is not yet advocated*
- *What is advocated is not yet acted on*
- *What is acted on is not yet completed*

§1 Recap, Intro & Overview

- Philosophy of (Computer) *Science*
- Power of Abstraction, Hardware vs. Math
- (Importance of) Asymptotic efficiency
- Sorting Example: specification & optimality
- Algorithm Analysis: How and Why?
- Course Overview and Administration
- Pedagogy & Teaching Philosophy