

# CECS 341: Theory of Computation I

## Overview

- Fundamental mathematical models of computation
  - What can and cannot be computed?
  - What computational models?
  - How quickly?
  - How much memory?
- Examples
  - a compiler detecting the presence of infinite loops?
  - a program checking whether or not two programs compute the same function?
  - a program determining the shortest program to compute a given function?

## Motivation and Overview

- Assess the difficulty of a given problem
- Measure the quality of a design
- Two major tasks for the theory of computation
  - Devise a universal model of computation
  - Characterize problems as solvable, efficiently solvable, simply solvable, . . .
- Tools for practice
  - grammars: programming languages
  - finite automata: string searching, pattern matching

## Automata, Computability, and Complexity

- *What are the fundamental capabilities and limitations of computers?*
- Automata theory
  - Definitions and properties of mathematical models of computation
    - \* finite automaton, push-down automaton, Turing machine
- Computability theory
  - distinguish solvable problems from unsolvable ones
- Complexity theory
  - *What makes problems computationally hard?*
  - Examples:
    - \* sorting problems vs. scheduling problems
    - \* games: tic-tac-toe, checker, chess, go

## History

- Late 19th to early 20th century
  - Questions about the nature of computing in the context of pure mathematics
  - Rigor and formalism in mathematical arguments
  - Some great mathematicians: Hilbert, Cauchy, Gödel
- 1930s and 1940s
  - Several universal models of computation (Church-Turing thesis)
  - Gödel and Kleen: the theory of partial recursive functions
  - Alonzo Church: lambda calculus
  - Alan Turing: Turing machine

## Recent Development

- 1950s and 1960s
  - From computability to complexity
  - Computing Pioneers: Alan Turing, John von Neumann
  - Stephen Cook and Richard Karp: NP-complete problems
- Recent years
  - Alternative models of computation
    - \* parallel computing
    - \* quantum computing
    - \* DNA computing
  - Proof theory
  - Randomized computing

## Assumed Background (Chapter 0)

- Sets / Sequences
- Functions / Relations
- Equivalence relations / Partitions
- Graphs
- Types of proof
  - Proof by construction
  - Proof by contradiction
  - Proof by induction