Welcome to CS 373

Theory of Computation

Spring 2010

Madhusudan Parthasarathy (Madhu)
madhu@cs.uiuc.edu

What is computable?

- Examples:
  - check if a number n is prime
  - compute the product of two numbers
  - sort a list of numbers
  - find the maximum number from a list

- Hard but computable:
  - Given a set of linear inequalities, maximize a linear function

    Eg. maximize $5x+2y$

    $3x+2y < 53$

    $x < 32$

    $5x - 9y > 22$

Theory of Computation

Primary aim of the course: *What is "computation"?*

- *Can we define computation without referring to a modern c computer?*

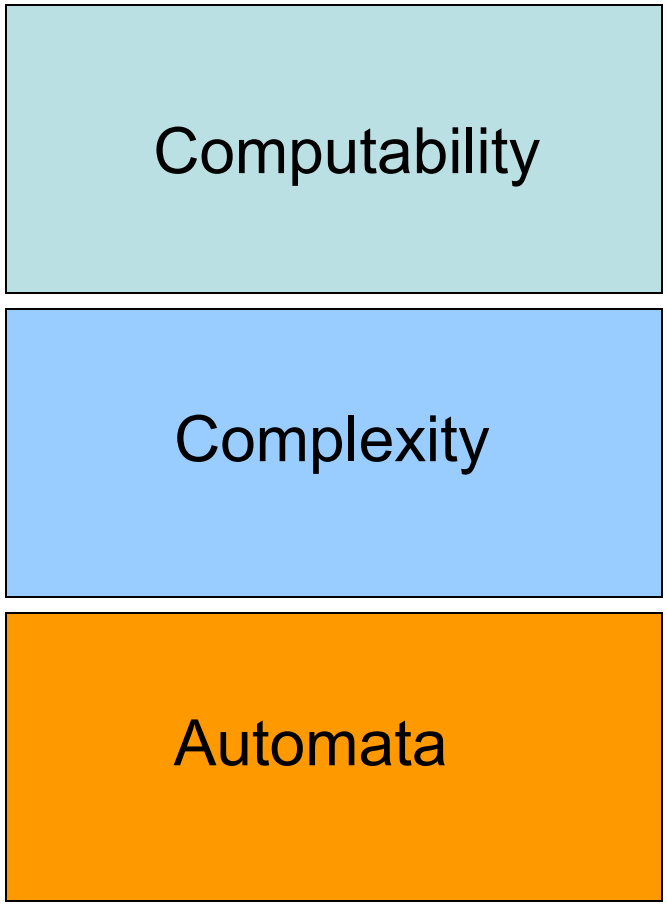
- *Can we define, mathematically, a computer?*
  (yes, Turing machines)

- *Is computation definable independent of present-day engineering limitations, understanding of physics, etc.?*

- *Can a computer solve any problem, given enough time and disk-space?*
  Or are they fundamental limits to computation?

In short, *understand the mathematics of computation*

Theory of Computation



- What can be computed?
- Can a computer solve any problem, given enough time and disk-space?

- How fast can we solve a problem?
- How little disk-space can we use to solve a problem

-What problems can we solve given really very little space? (constant space)

Theory of Computation

What problems can a computer solve?

Computability

Complexity

Automata

Not all problems!!!

Eg. Given a C-program, we cannot check if it will not crash!

Verification of correctness of programs is hence impossible!

(The woe of Microsoft!)

Theory of Computation

What problems can a computer solve?

Computability

Complexity

Automata

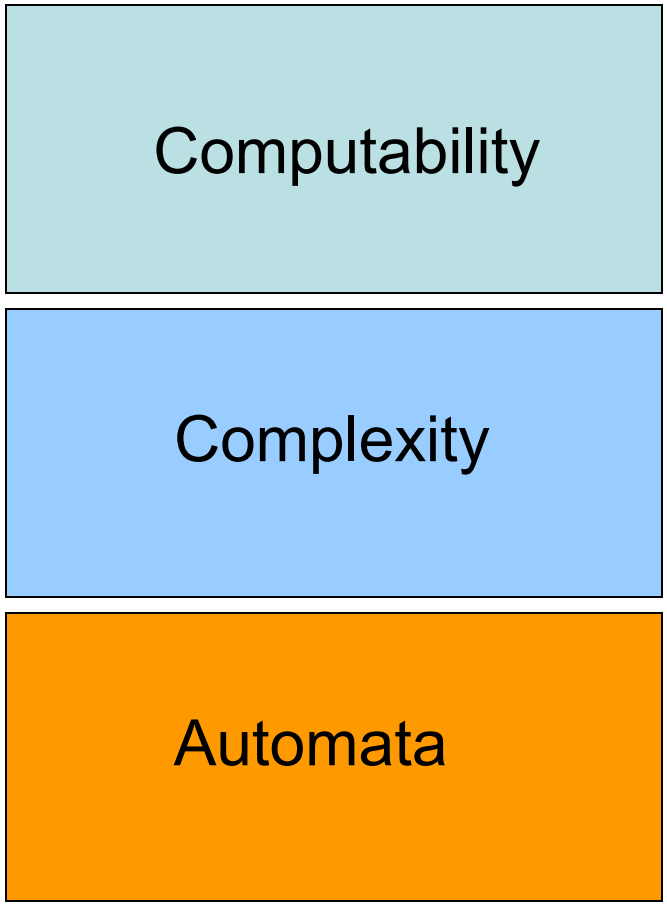
Even checking whether a C-program will halt/terminate is not possible!

```
input n;
assume n>1;
while (n !=1) {
   if (n is even)
     n := n/2;
  else
     n := 3*n+1;
}
```

No one knows whether this terminates on on all inputs!

17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.

Theory of Computation



How fast can we compute a function?
How much space do we require?

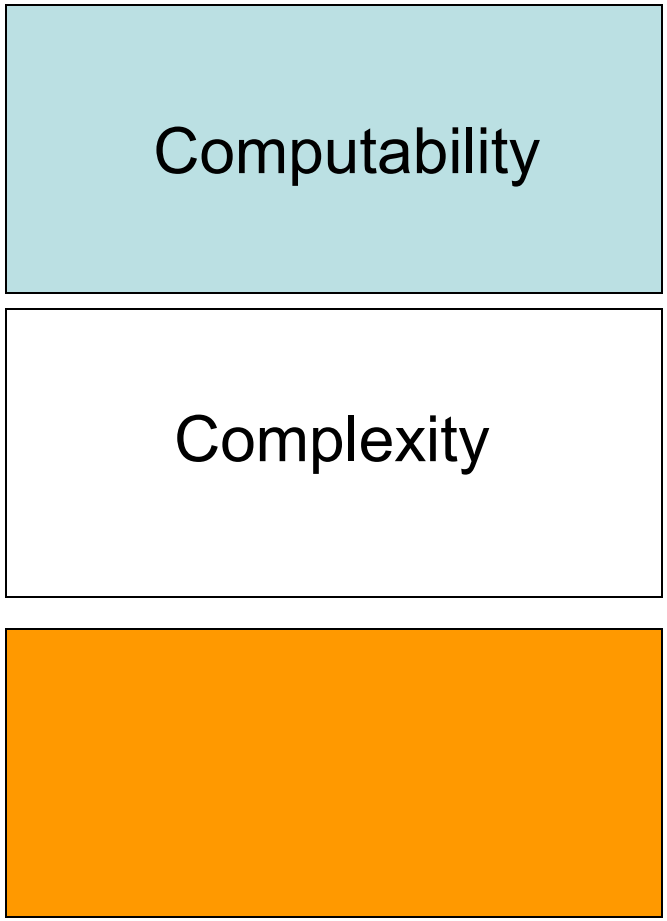
- **Polynomial time computable**
- **Non-det Poly Time (NP)**
- **Approximation, Randomization**

Functions that cannot be computed fast:

- **Applications to security**
  - **Encrypt fast,**
  - **Decryption cannot be done fast**

- **RSA cryptography,**
  web applications

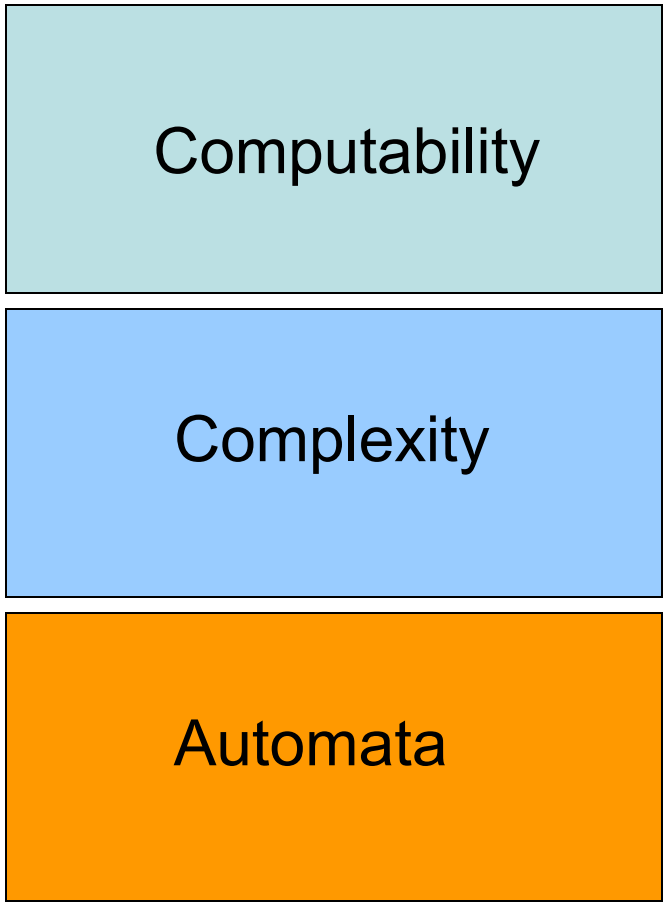
Theory of Computation



Machines with finite memory:--
traffic signals, vending machines
hardware circuits

Tractable.
Applications to pattern matching,
modeling,
verification of hardware, etc.

Theory of Computation

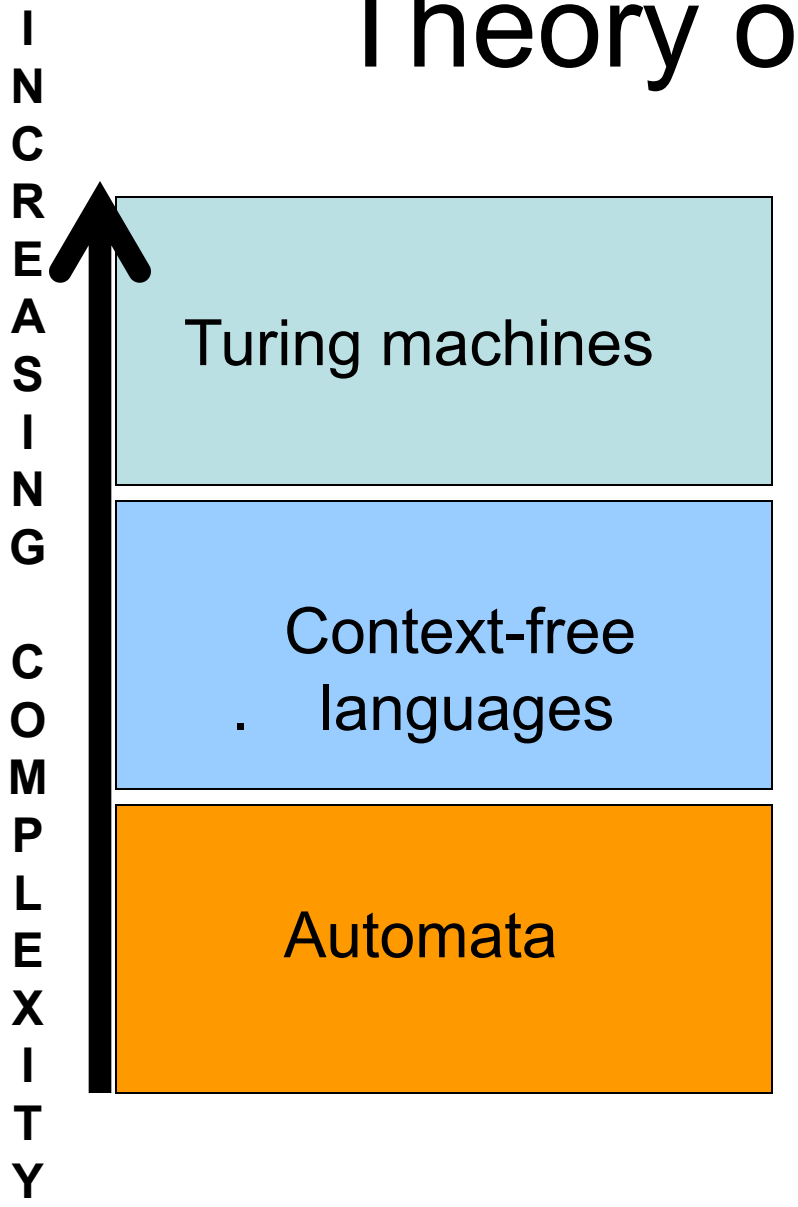


What can we compute?
-- Most general notions of computability
-- Uncomputable functions

What can we compute fast? CS473!
-- Faster algorithms, polynomial time
-- Problems that cannot be solved fast:
* Cryptography

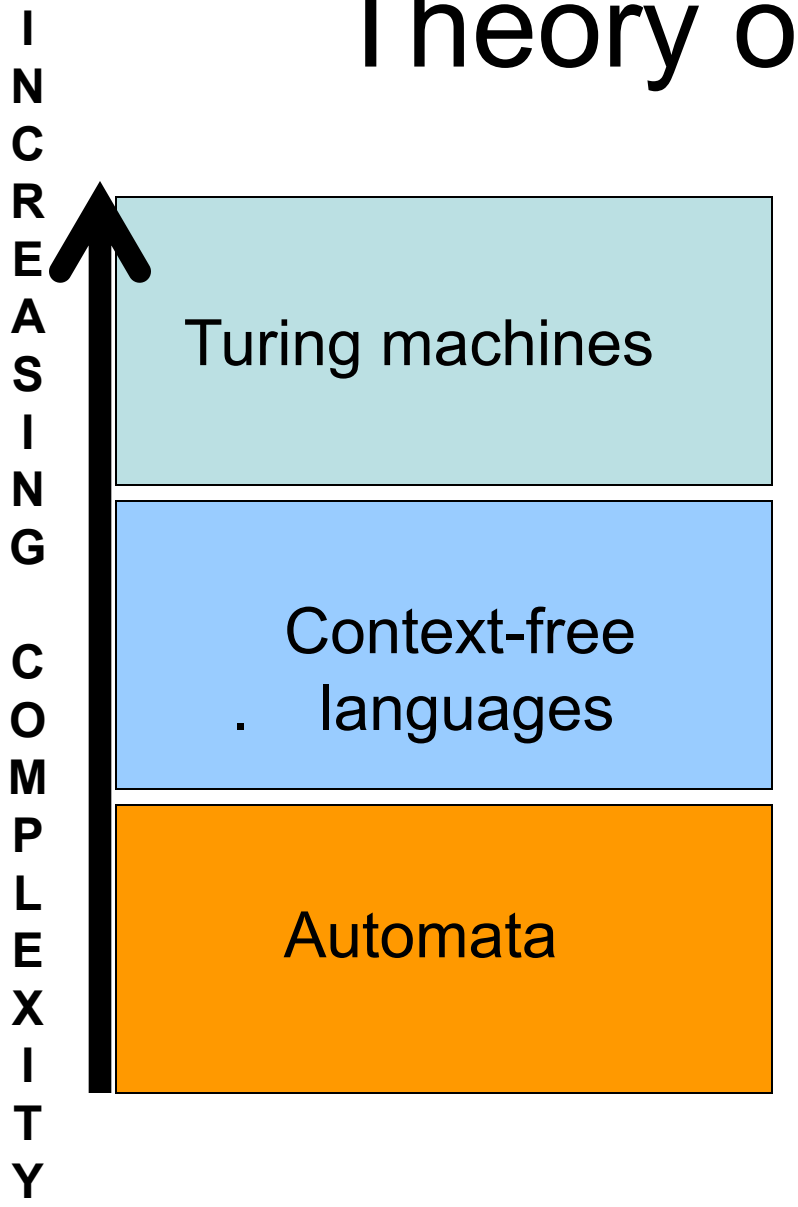
What can we compute with very little space?
-- Constant space (+stack)
* String searching, language parsing, hardware verification, etc.

Theory of Computation



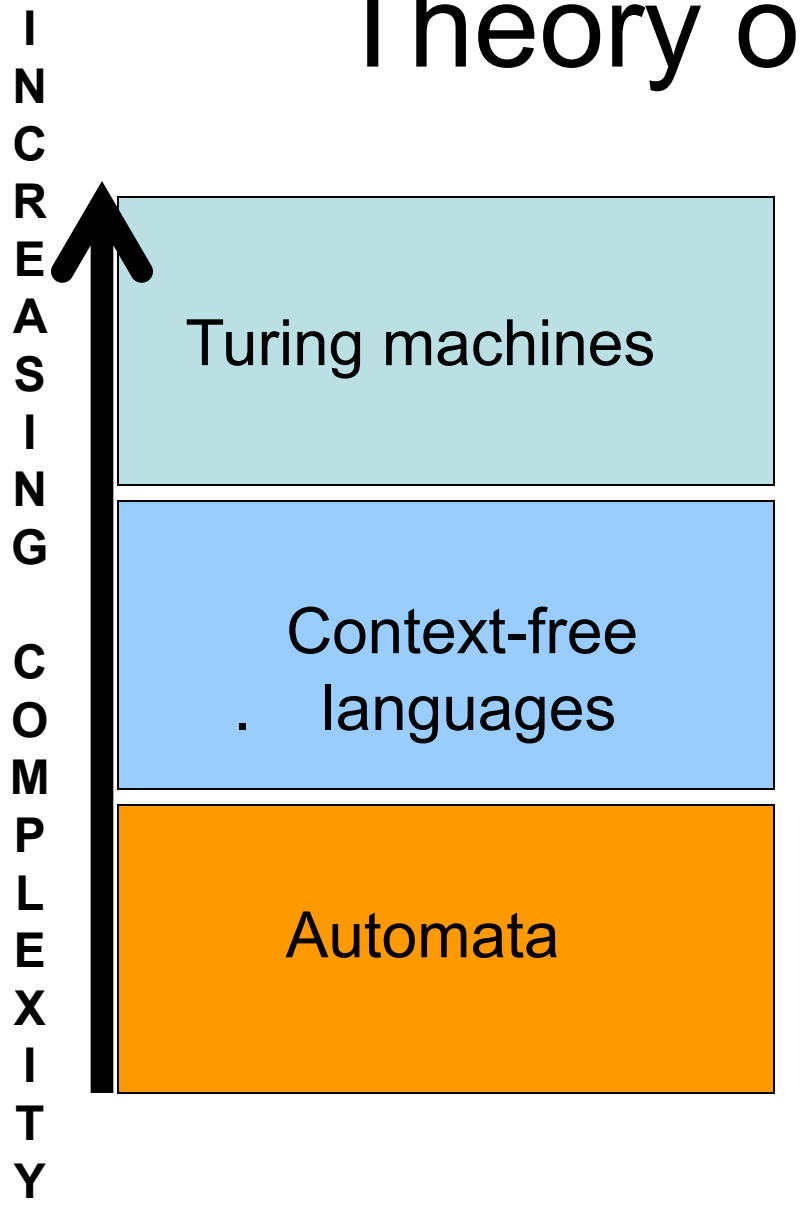
Automata:
--- Foundations of computing
--- Mathematical methods of argument
--- Simple setting

Theory of Computation



Context-free languages
--- Grammars, parsing
--- Finite state machines with recursion (or stack)
--- Still a simple setting; but infinite state

Theory of Computation

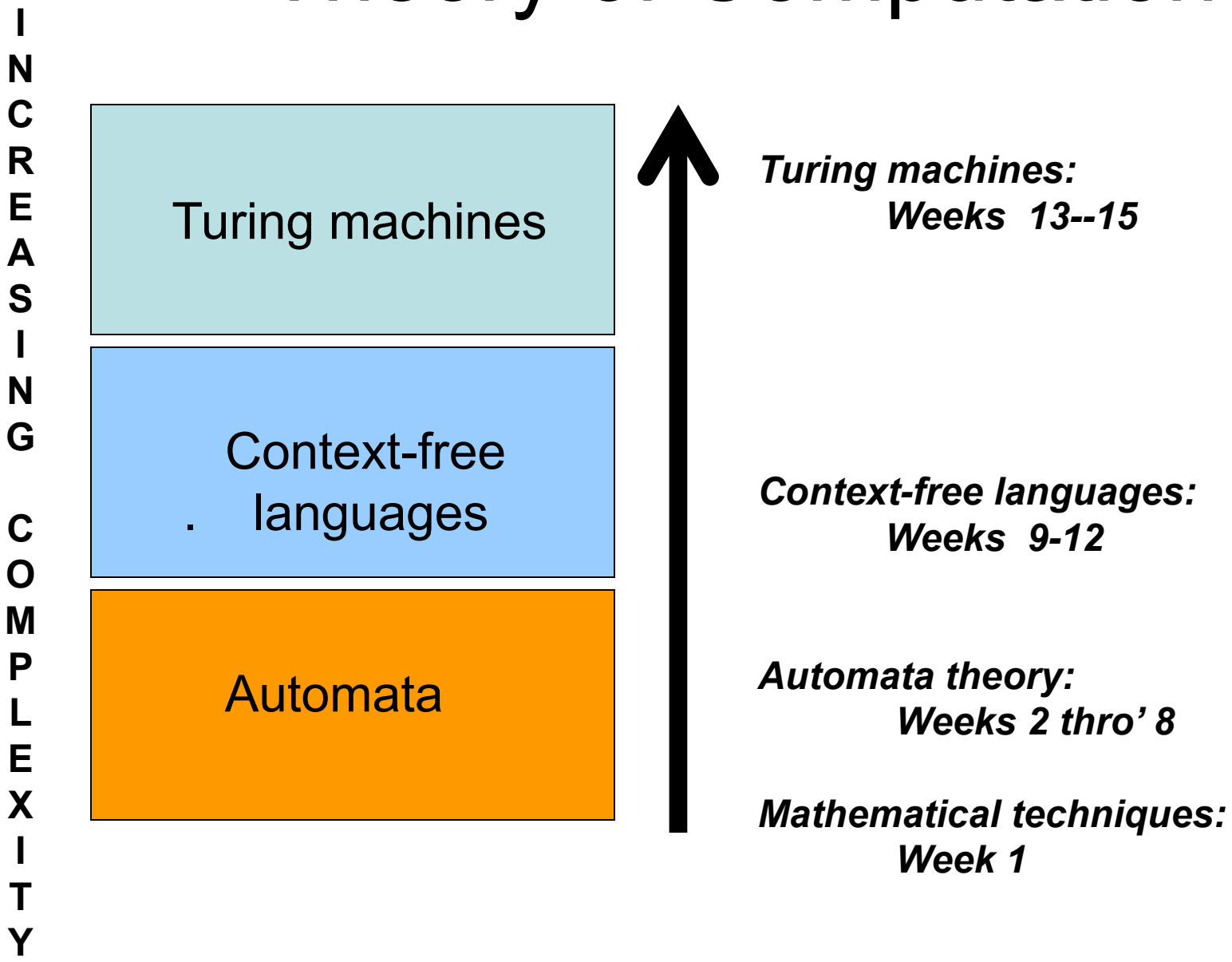


Turing machines (1940s):
-- The most general notion of computing
-- The Church-Turing thesis
-- Limits to computing:
Uncomputable functions

Motivation from mathematics:

- Can we solve any mathematical question *methodically*?
- Godel's theorem: NO!
- "Even the most powerful machines cannot solve some problems."

Theory of Computation



Kurt Gödel

- Logician extraordinaire
- Hilbert, Russel, etc. tried to formalize mathematics
- “Incompleteness theorem” (1931)
  - Cannot prove consistency of arithmetic formally
  - Consequence: unprovable theorems

Kurt Godel: 1906 - 1978

Alonzo Church

First notions of computable functions

First language for programs

-- lambda calculus

-- formal algebraic language

for computable functions

Alonzo Church:
1903 - 1995

Alan Turing

- "father of computer science"
- Defined the first formal notion of a computer (Turing machine) in 1936: "*On Computable Numbers, with an Application to the Entscheidungsproblem*"
- Proved uncomputable functions exist ("halting problem")

- Church-Turing thesis: all real world computable functions are Turing m/c computable

- Cryptanalysis work breaking Enigma in WW-II

Alan Turing: 1912 - 1954

Noam Chomsky

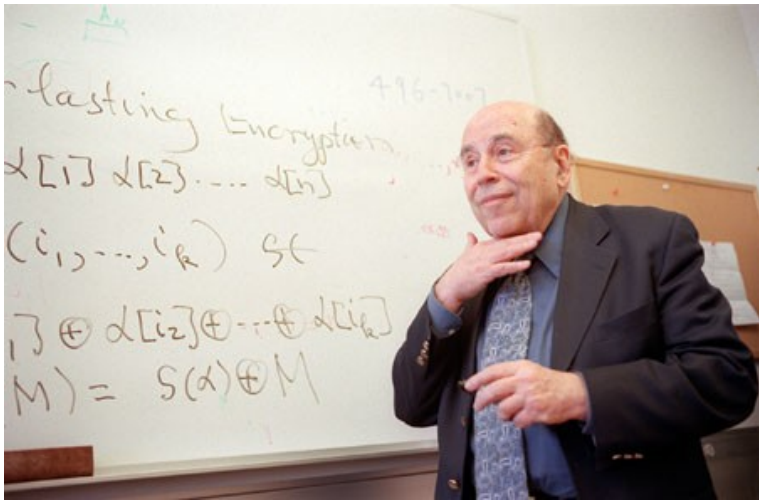
- Linguist ; introduced the notion of formal languages arguing generative grammars are at the base of natural languages
- Hierarchy of formal languages that coincides with computation
- Eg. Context-free grammars capture most skeletons of prog. languages

Noam Chomsky: 1928-

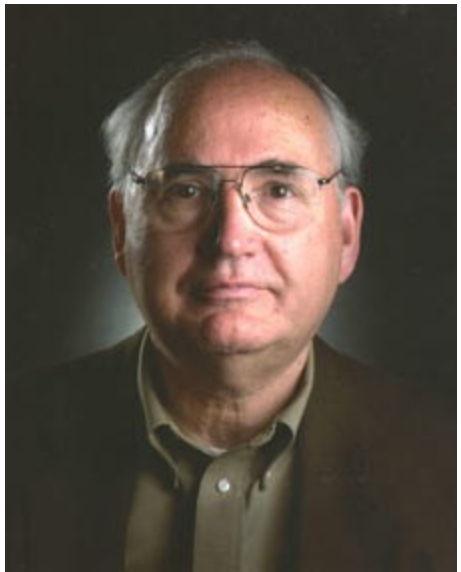
"Logical Structure of Linguistic Theory" (1957)

Automata theory

- *Automata: machines with finite memory*



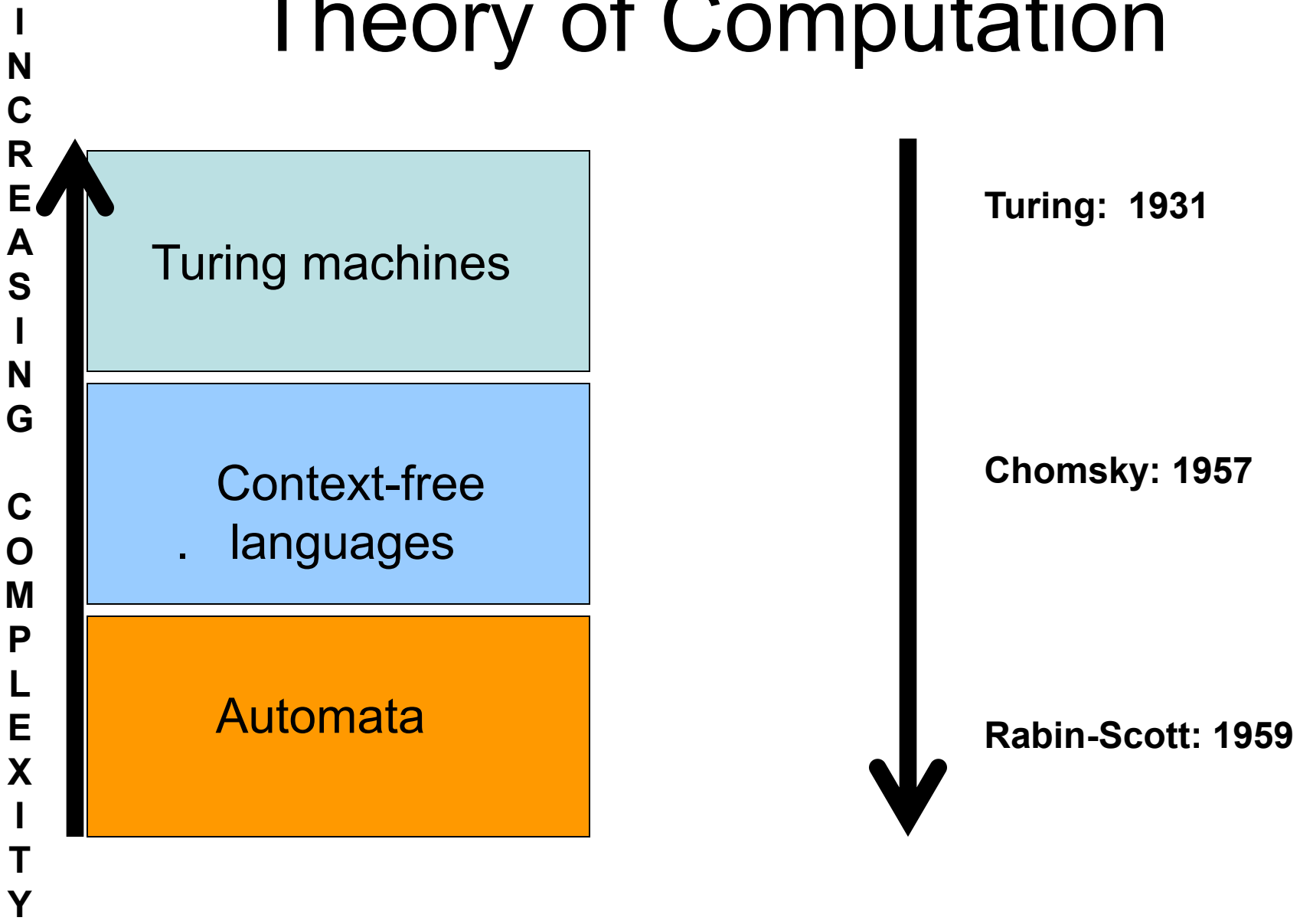
- *"Finite Automata and Their Decision Problem"*
  - Rabin and Scott (1959)



- *Introduced nondeterministic automata and the formalism we still use today*

- *Initial motivation: modeling circuits*
- *Turing Award (1976)*

Theory of Computation



Goals of the course

- To understand the notion of “computability”
- Inherent limits to computability
- The tractability of weaker models of computation
- The relation of computability to formal languages

- Mathematics of computer science
  - Rigor
  - Proofs

A result you would know at the end...

- *Proving that it is impossible to check if a C program will halt.*

- Formal proof!

  You can convince a friend using a paper-napkin argument

- No computer *ever* will solve this problem

  (not even a quantum computer)

Textbook

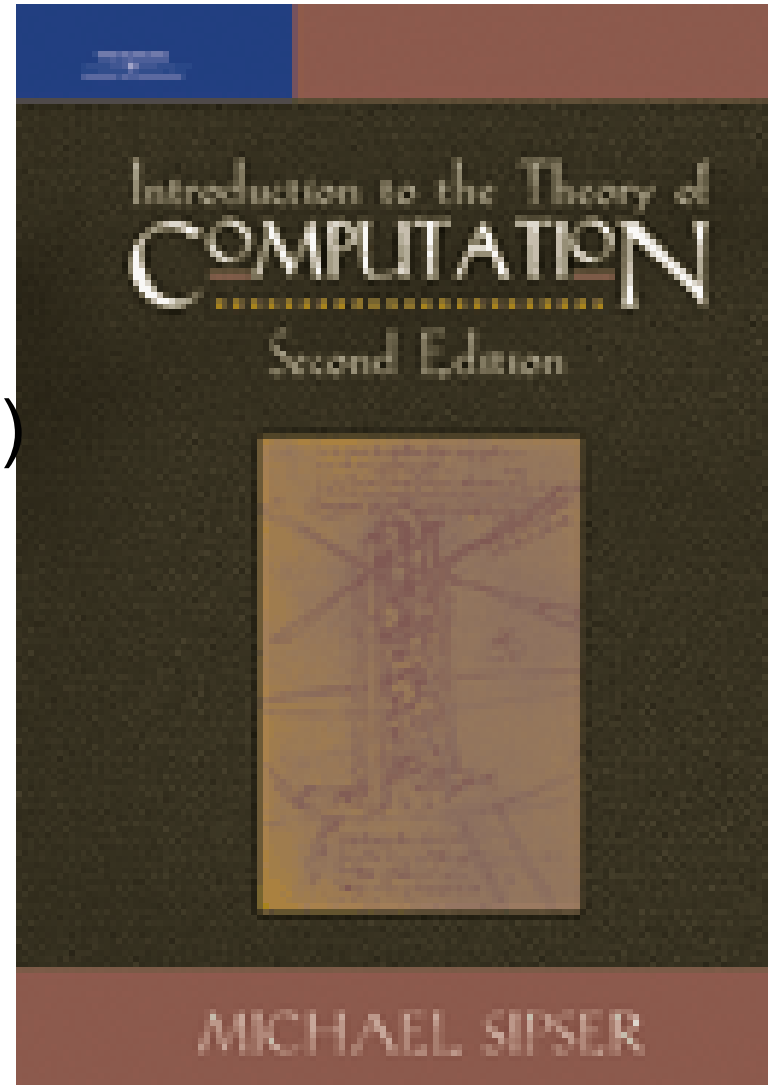
Michael Sipser

Introduction to Theory of Computation

(2nd ed; 1st ed may be ok)

I will announce chapter readings that you must read before class.

Hopcroft-Ullman



Course logistics

- Tu/Thu 2:00pm – 3:15pm Lectures in SC 1105.
- Discussion sections (all in SC 1111): by TAs

  Wed 10:00 am - 10:50 am

  Wed 11:00 am - 11:50 am

  Wed 12:00 pm - 12:50 pm

  Wed 2:00 pm - 2:50 pm

  Wed 3:00 pm - 3:50 pm

No discussion section tomorrow!

- Announcements (homework posting announcements, discussions, hints for homework, corrections/clarifications):

  Newsgroup: class.cs373

Teaching assistants

- Reza Zamani zamani@uiuc.edu
- Dmytro Suvorov suvorov1@illinois.edu
- Xiaokang Qiu qiu2@illinois.edu
- Office hours: will be posted; could change over semester (more during exams)

Furloughs

- University has announced furloughs this semester
- This *should not* affect you in any way for this course

Problem Sets

- Homeworks every week or every other week;
  (homework assigned for two weeks have more problems ☺)
  Posting dates and hand-back dates will be posted.
- Write each problem on separate sheet of paper. (for distributed grading)
  Don't staple them!
- Homework can be done in groups of at most three people.
- However, each student must hand in their own homework
  (no group submissions; must clearly write your group members)
- Work in a group, but think and try to solve each problem yourself!
  - Don't "distribute the problems within the group"
    If you do, it will hurt you in your exams (which account for 70%)
- Simple late-HW policy: **Late homeworks will not be accepted.**
- There may be additional "quizzes" (15min-30min tests) at discussion sections and online as well.

Problem Sets

~130 students and 5 problems a week, our team has to grade 650 problems each week! (~6500 for semester)

Help us to do this work efficiently:

- Read and follow the homework course policy
- Don't write junk; if you simply answer your question with "I don't know", or something to that effect, *and* nothing else, gives you automatically 20% credit (except for *extra* credit problems)
- Read and follow honesty and integrity policy
- Even if you find a solution elsewhere (you can use any outside source you please), but you *must* cite this source, and write the solution in your own words. Remember that these sources will vanish when exams arrive.

Grading

- Two midterms - 20% each
- Final exam - 30%

} 70%

- Homework and Quizzes - 25%
  (least scored HW not counted)

- Attendance to
  discussion sections - 5%

Curve

- Raw numerical scores tend to run low in theory classes; letter grades will primarily be decided based on relative ranking within the class, which will be approximately:

Class Percentile	Grade
95 %	A+
85 %	A
80 %	A--
70 %	B+
60 %	B
50 %	B--
40 %	C+
30 %	C
20 %	C--
15 %	D+
10 %	D
5 %	D-
Determined student by student	F

Curved grading

- At least four times during the semester,

  (perhaps more often), we will release your current grade according to the curve.

- An F-score is done extremely carefully, examining homework scores and final exam manually. We give an F only to students who stopped even attempting to do the work, or have understood very little of the material.

  Ideally, we'd like everyone either drop the course early on, or else pass it.

My lectures

- I will use a tablet PC; all class lecture slides will be posted online.
- Additional resources (on course webpage)
  - Lecture notes from Spring'08 (Sariel and me)
  - Lecture notes (slides) from Fall'08
  - Review notes on main results you should learn/know (by me)
  - Old homeworks/solutions online
  - Probably too many resources!....

Honors?

- Honors students will do extra problems and a project.

- Please contact me after class if you intend taking this course as an honors course.

How to do well…

- This is essentially a math course:
  - you must learn the concepts well; if you don't there's almost no chance of success
  - if you do learn the concepts, there is very little else (facts, etc.) to learn; you can do really well!
  - You must do problems. There's no replacement for this.
  - Attending lectures is highly adviced!
    - It will be very hard to learn the concepts by yourself or from textbook.
  - Don't postpone learning; you will not be able to "make up" later. Topics get quickly hard.
  - Come regularly to discussion sections; you will learn a lot by working out problems and learn from fellow students

How to do well…

- Come to office hours!!
  - We are here to help you learn and do well.

– Check out http://www.cs.uiuc.edu/class/sp10/cs373/ in the next few days

– Homework#0 will be handed out this week (probably Friday)