2021

Theory of Computation

Kun-Ta Chuang
Department of Computer Science and Information Engineering
National Cheng Kung University


年度最強災難強片
八月三十日全台同步上映
開學
BACK TO SCHOOL
台灣教育部嘔心力作
全台兩百萬師生含淚演出

Outline

Preliminaries

Course Objectives

Course Contents

Preliminaries

- **Course Information**
  - Course Name: Theory of Computation
    (計算理論)
  - Time: Wednesday 2:10 pm ~ 5:00 pm
  - Course Website: ncku moodle

Preliminaries

- **Instructor**
  - Name: Kun-Ta Chuang (莊坤達)
  - E-mail: ktchuang@mail.ncku.edu.tw
  - Office Hours: Thursday 10:00 am ~ 12:00 am
  - Office Location: CSIE 6F 608
- **Teaching Assistants**
  - 丁羅邦芸
  - 張財實
  - 陳冠廷
  - 孫毅夫
  - 郭哲瑋
  - EMAIL TO ncku.toc.ta@netdb.csie.ncku.edu.tw

Preliminaries

- Startup -- Youthwant (1999-2001)
- Startup -- UniPattern (2002-2004)
- Join Synopsys (2006 – 2011)
  - Served as a Senior Engineer (國防役)
- Synopsys is the world leader in EDA
- Division of Synopsys Design-Rule Check
  - R&D teams in US/Taiwan (Taiwan: ~40 engineers)
  - Customers includes
    - IC gaints, e.g., Intel, Samsung, TI, Nvidia, Qualcomm, Broadcom, Novatek;
    - Major IDMs, e.g., Panasonic, Toshiba;
    - Key foundries, TSMC, UMC, Global Foundries

SYNOPSYS®

current or previous clients, and I do not hold equity positions in any of them.

The Best Software Companies To Work For

The following table ranks the PWC List of Top 100 Global Software Leaders by the percentage of employees who would recommend their employer to a friend. Agfa HealthCare, CompuGROUP Holding, Constellation Software, DATEV, Hexagon , TOTVS and Visma aren't included in this table as the number of reviews on Glassdoor are very small.

Company	% of employees who would recommend this company to a friend	% of Employees who approve of the CEO as of July 12, 2013 on Glassdoor
Google	90%	95%
InterSystems	88%	93%
Citrix Systems	85%	92%
SAP	84%	94%
Adobe	84%	70%
SAS	84%	87%
Intel	83%	92%
Synopsys	83%	96%
Informatica	83%	92%
NetApp	82%	91%
Apple	81%	93%
Mentor Graphics	81%	86%
Bentley Systems	80%	92%
Intuit	79%	91%
Red Hat	79%	94%
Teradata	78%	80%
Microsoft	77%	47%
Ericsson	77%	88%

媽,我在這裡

Louis Columbus
Contributor
+ Follow (191)

I'm serving as Product Marketing Manager for Plex Systems, a leading provider of Cloud-based ERP systems for manufacturers where my responsibilities include new product introductions, messaging, marketing strategy, business development, and competitive strategy. Previous positions include senior analyst at AMR

+ show more

The author is a Forbes contributor. The opinions expressed are those of the writer.

LOUIS COLUMBUS' POPULAR POSTS

The Best Enterprise Software Companies And CEOs To Work For In 2013 129,432 views

Hype Cycle for Cloud Computing Shows Enterprises Finding Value in Big Data, Virtualization 43,886 views

The Best Cloud Computing Companies and CEOs to Work For in 2013 41,115 views

Gartner Hype Cycle for CRM Sales, 2012: Sales Turns to the Cloud for Quick Relief 39,974 views

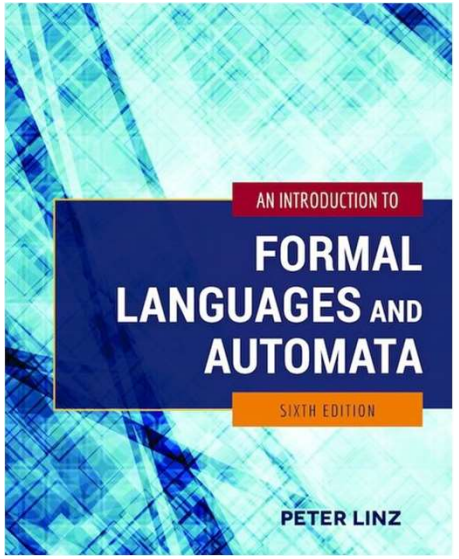
Cloud Computing and Enterprise Software Forecast Update, 2012 38,584 views

MORE FROM LOUIS COLUMBUS

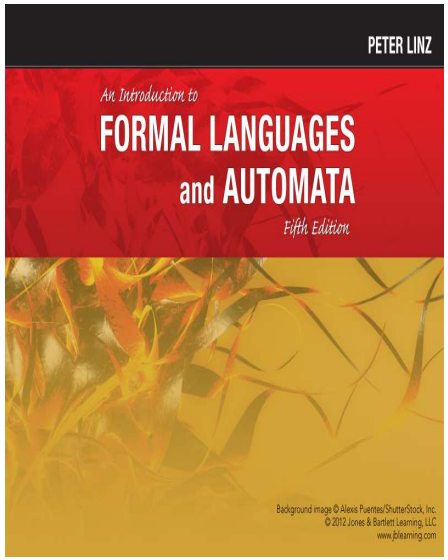
Preliminaries

Preliminaries

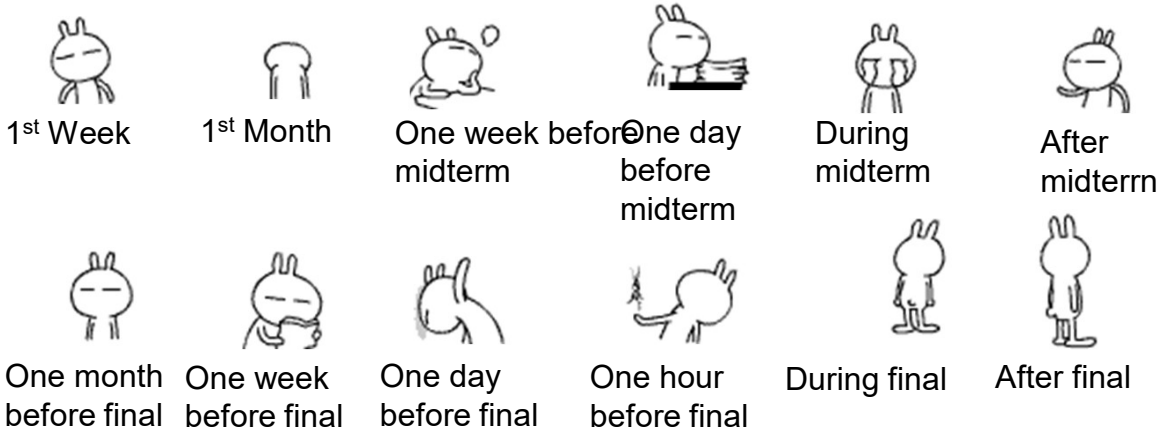
- **Text Book**
  - Name: An Introduction to Formal Languages and Automata, 6th Edition (5th Edition)
  - Author: Peter Linz (UC Davis)
  - Publisher: Jones and Bartlett Publishers, 2016 (2011)



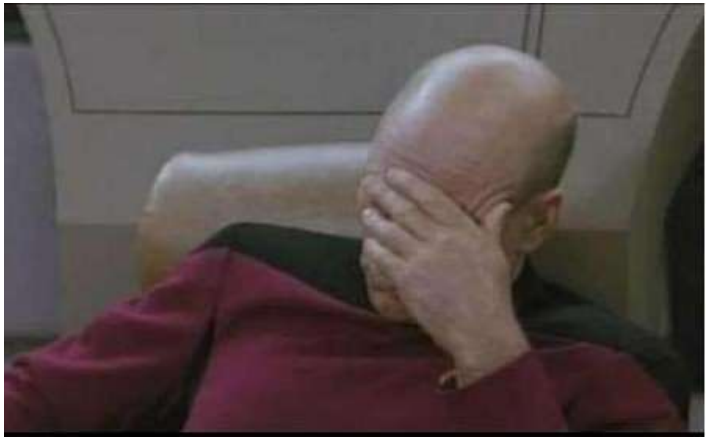
- **Reference Book**
  - Name: Introduction to Automata Theory, Languages and Computation, 3rd Edition
  - Author: John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman
  - Publisher: Addison Wesley, 2006



Course Schedule



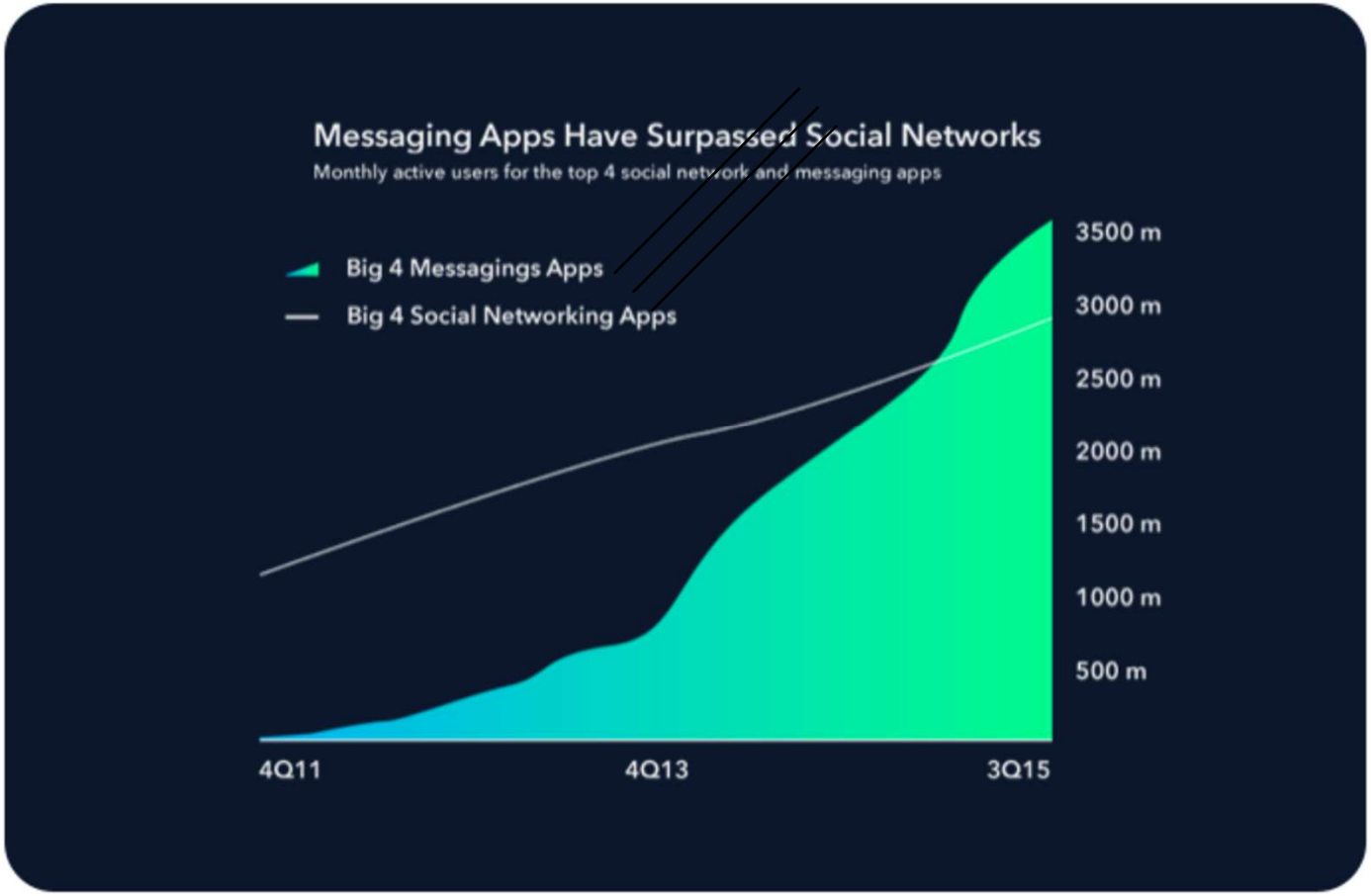
- **Grading:**
  - Homework assignments: 30%
  - Midterm Examination: 25%
  - Final Examination: 30%
  - Program Exercise (1 HWs): 15%
- **Penalty for late submission:** 20% per day.



Program HW

- Write a Chat Bot

為什麼我們應該關注 Chatbot? 它難道不是只是個聊天介面嗎？讓我們用數據來告訴你，為什麼使用者重視它。目前使用聊天通訊軟體的人數已經超越了使用社群平台的人數了！光是 Facebook Messenger 及 WhatsApp 的使用者便已經超越了17億人，相關研究也顯示，使用者每天有超過90%以上的時間，花在使用聊天軟體及平台上。



Outline

Preliminaries

Course Objectives

Course Contents

Course Objectives

- The study of the theory of computation has several objectives, most importantly
  - To familiarize students with the foundations and principles of computer science
  - To teach material that is useful in subsequent courses
  - To enhance the problem solving capabilities
  - To strengthen students' ability to carry out formal and rigorous mathematical arguments

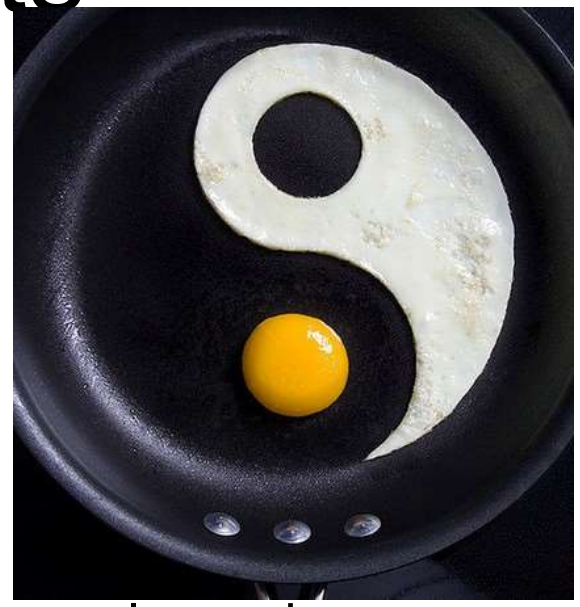
Outline

Preliminaries

Course Objectives

Course Contents

Course Contents



Theory of computation, includes

A. Automata theory
B. Formal languages and grammars
C. Computability
D. Complexity

A and B: Model of computation

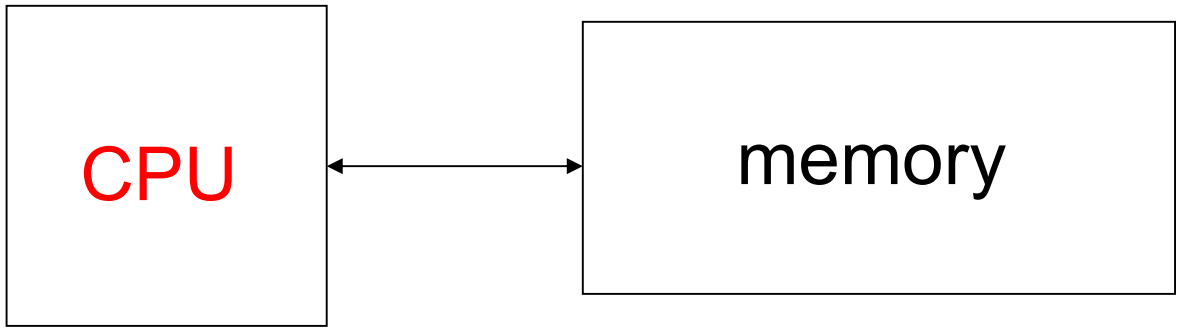
C: Under which assumptions on our computer can we solve a given problem?

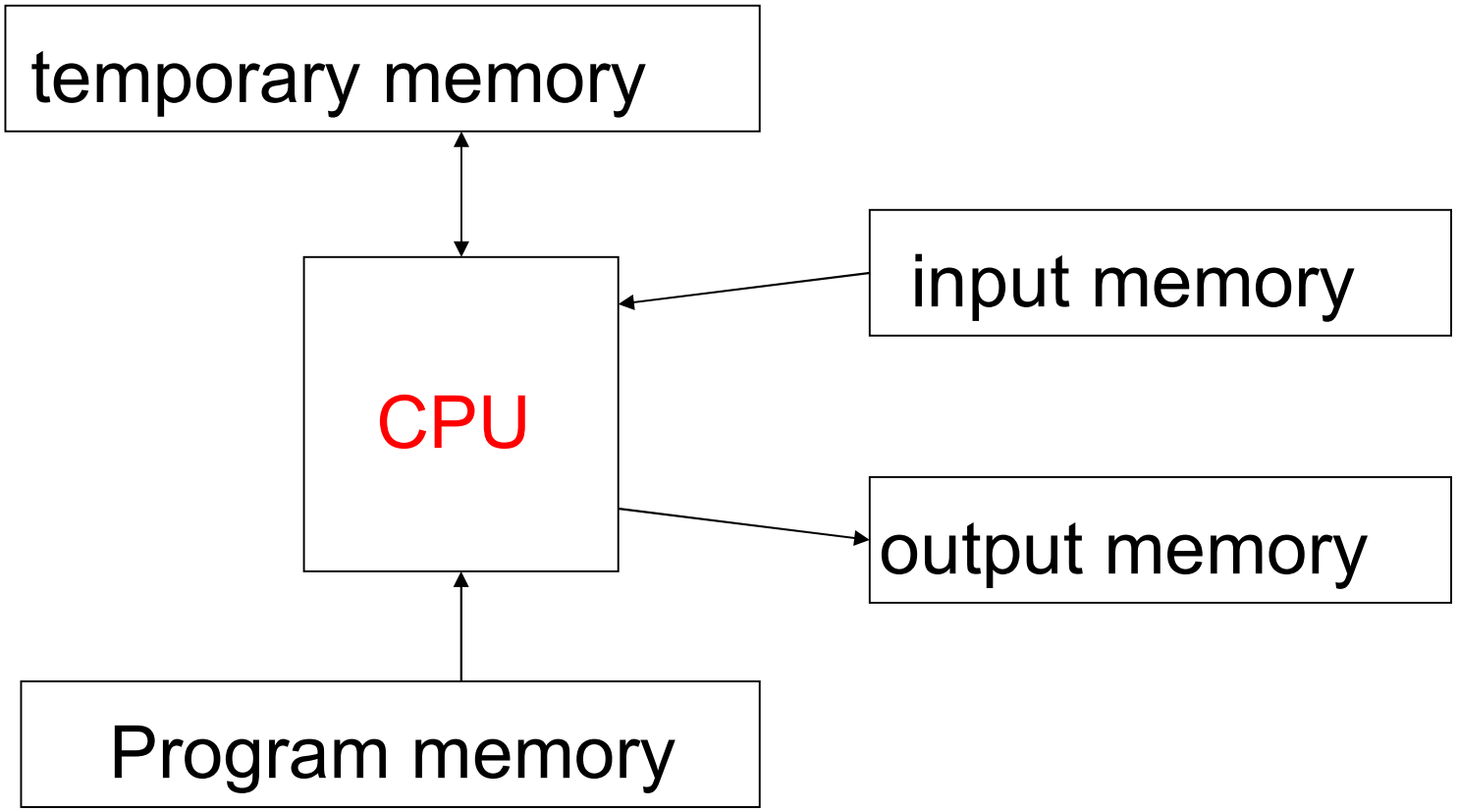
D: Assuming that we can solve a problem, how hard is it to solve?

Think: memory and time requirements.

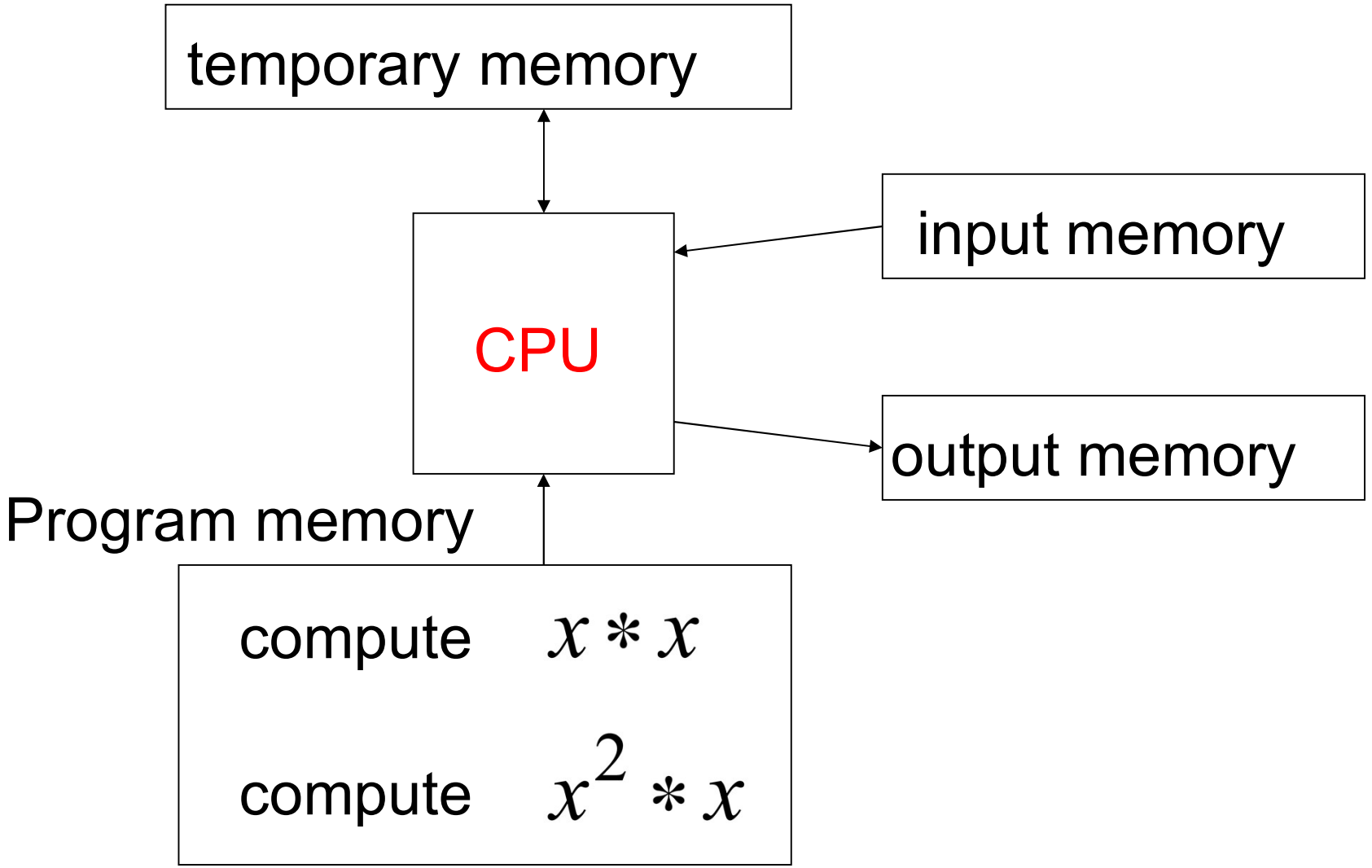
We will study various automata, see how they are related to languages and grammars, and investigate what can and cannot be done by digital computers

Computation

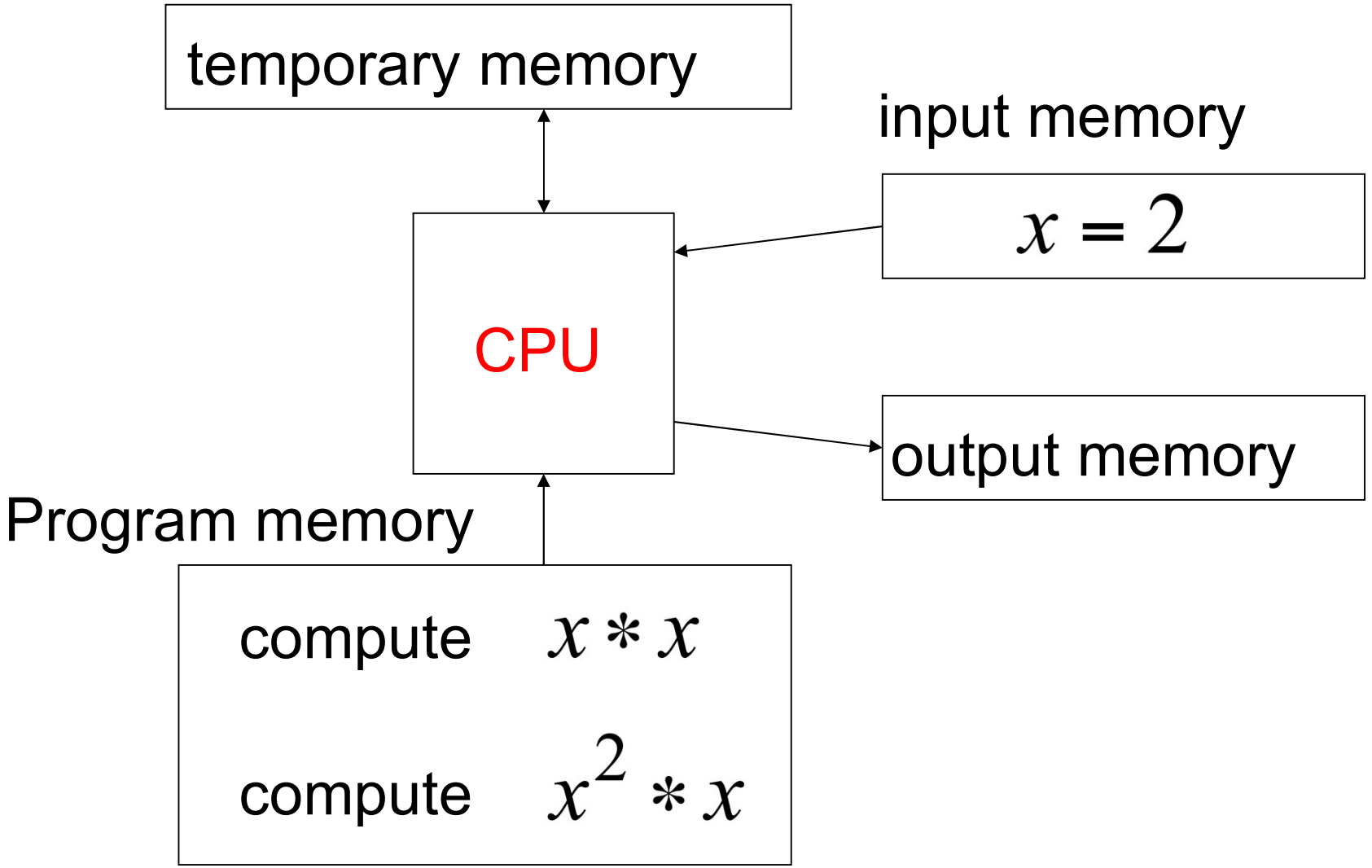



temporary memory
input memory
CPU
output memory
Program memory

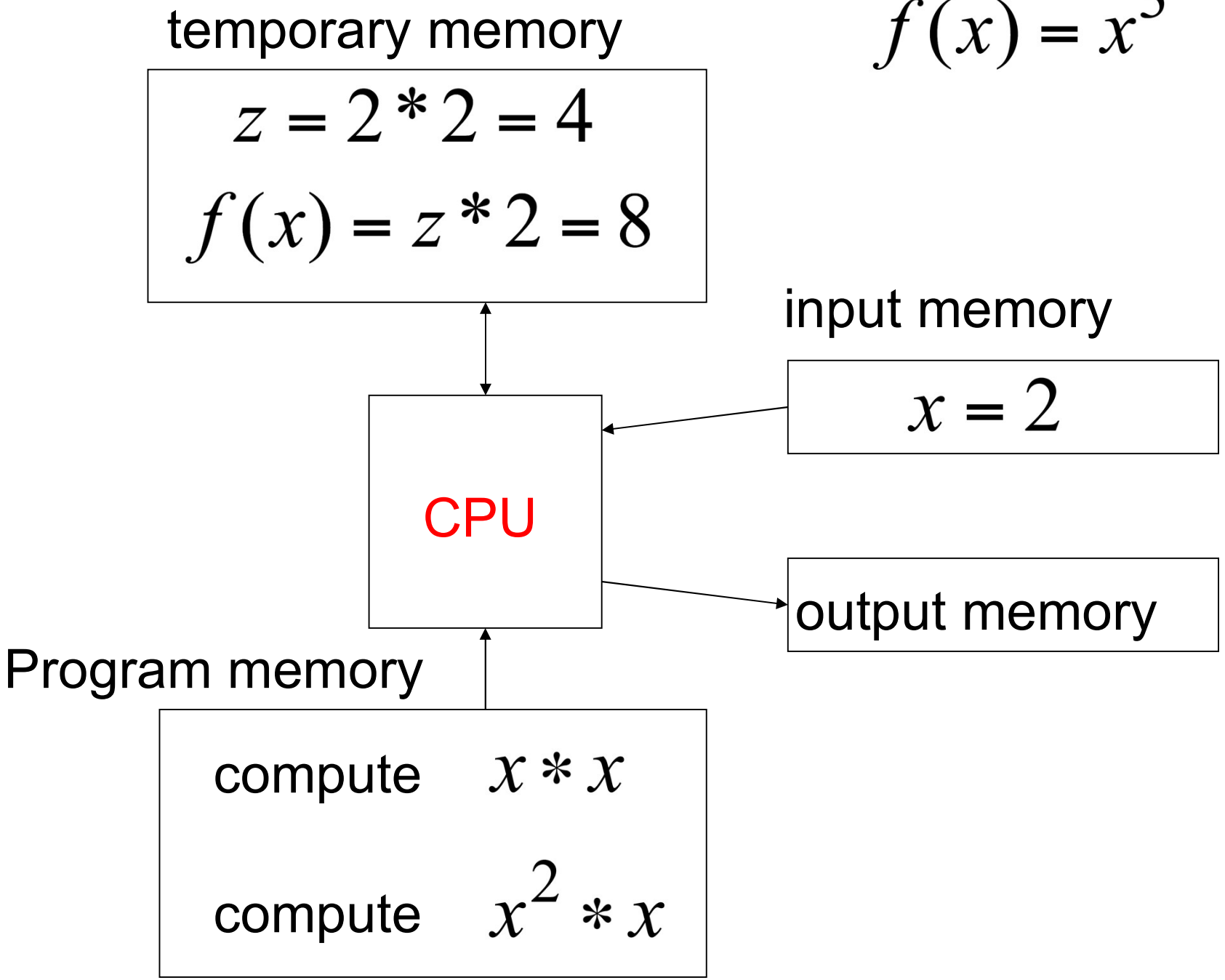
Example: $f(x) = x^3$



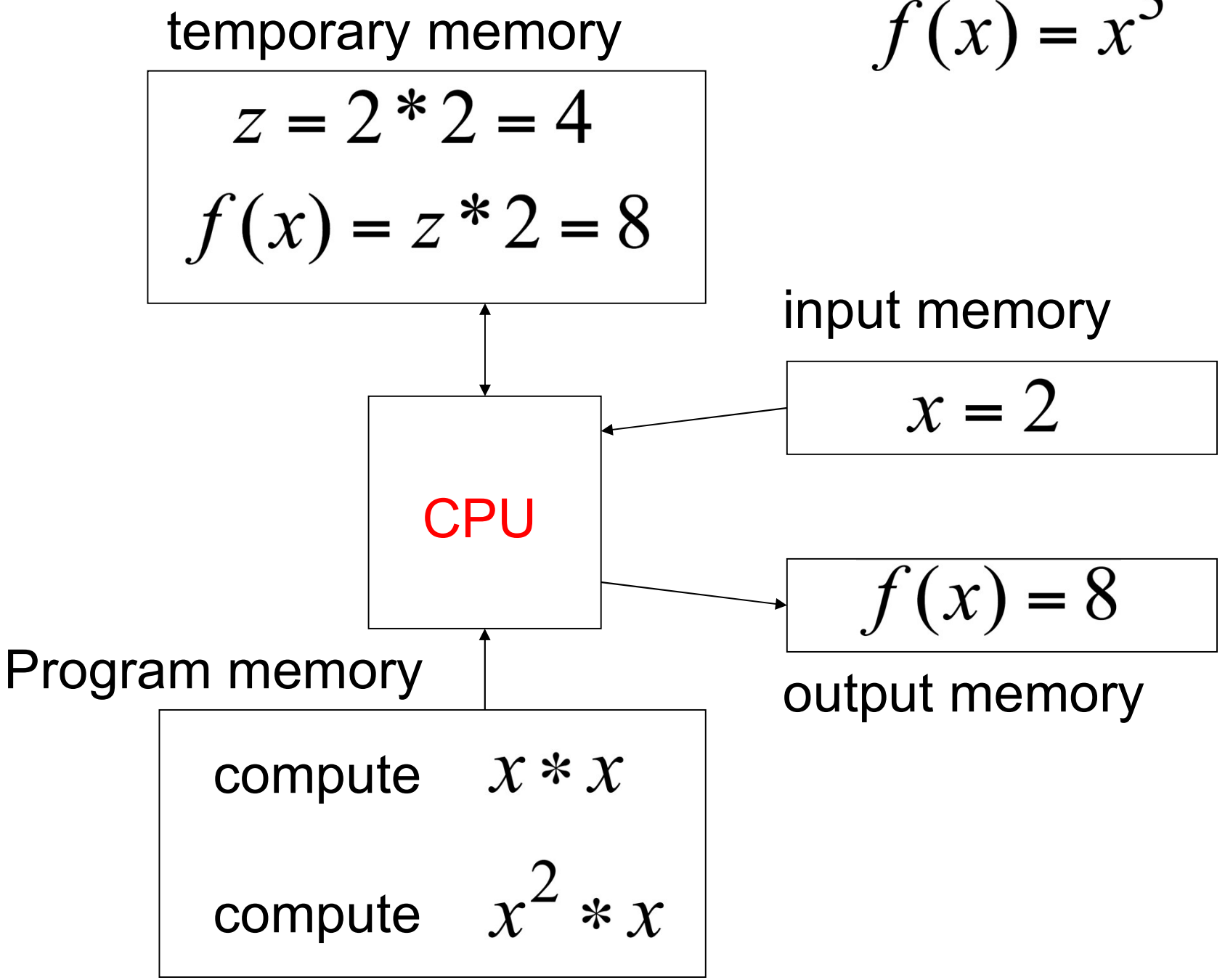
$$f(x) = x^3$$


temporary memory
input memory
$x = 2$
CPU
output memory
Program memory
compute $x * x$
compute $x^2 * x$

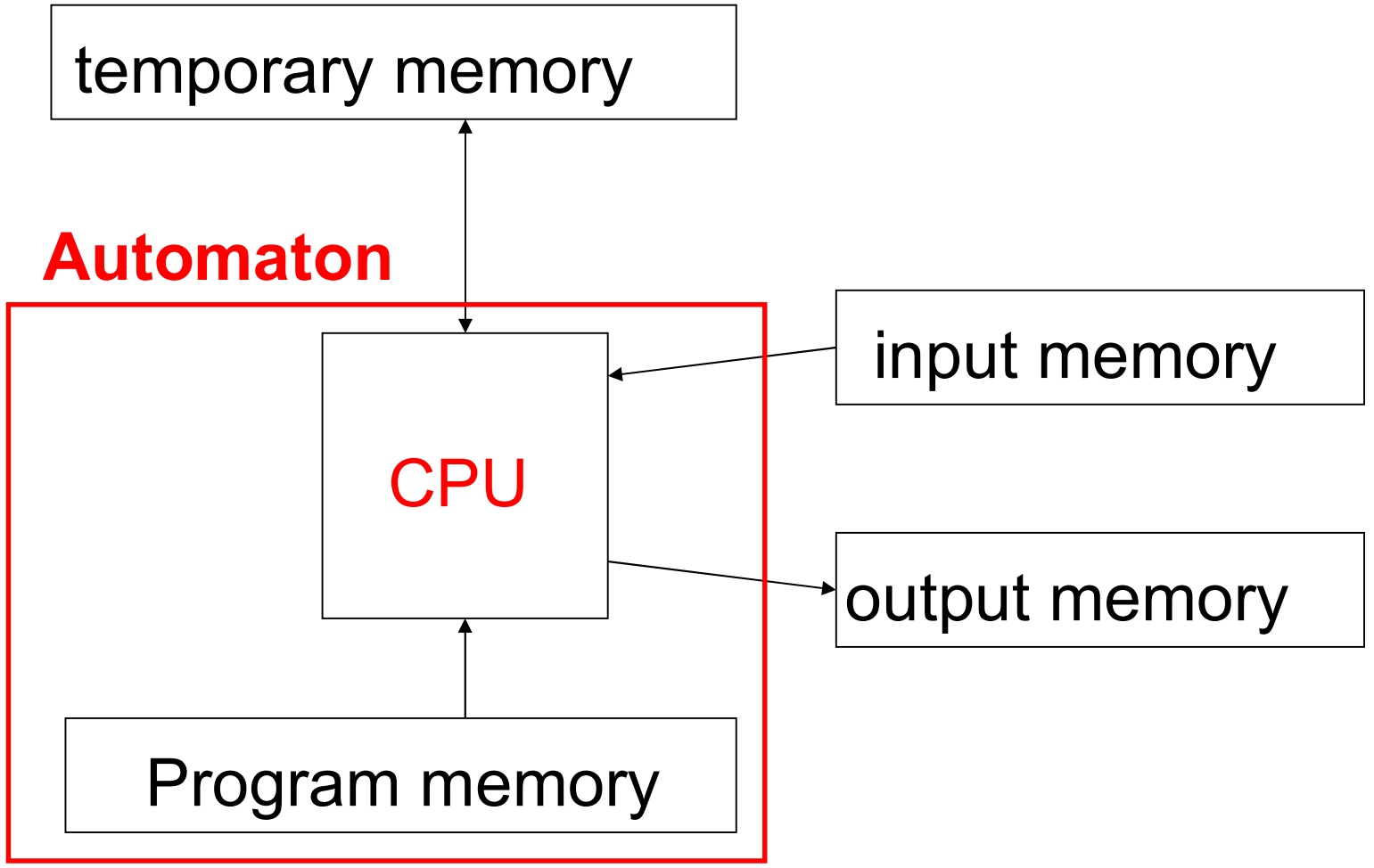
$$f(x) = x^3$$


temporary memory
$z = 2 * 2 = 4$
$f(x) = z * 2 = 8$
input memory
$x = 2$
CPU
output memory
Program memory
compute $x * x$
compute $x^2 * x$

$$f(x) = x^3$$


temporary memory
$z = 2 * 2 = 4$
$f(x) = z * 2 = 8$
input memory
$x = 2$
CPU
$f(x) = 8$
output memory
Program memory
compute $x * x$
compute $x^2 * x$

Automaton

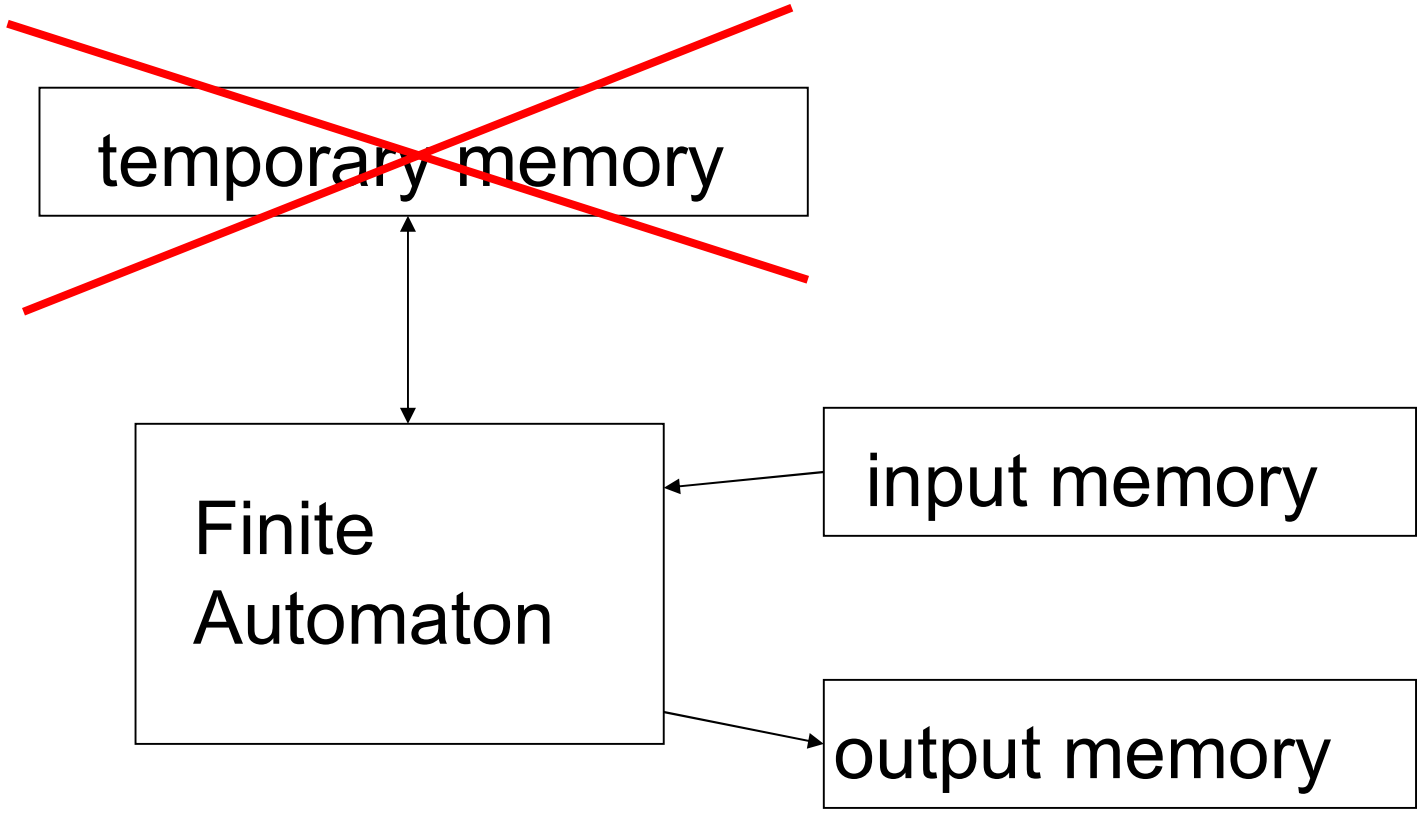

temporary memory
Automaton
input memory
CPU
output memory
Program memory

Different Kinds of Automaton

Automata are distinguished by the temporary memory

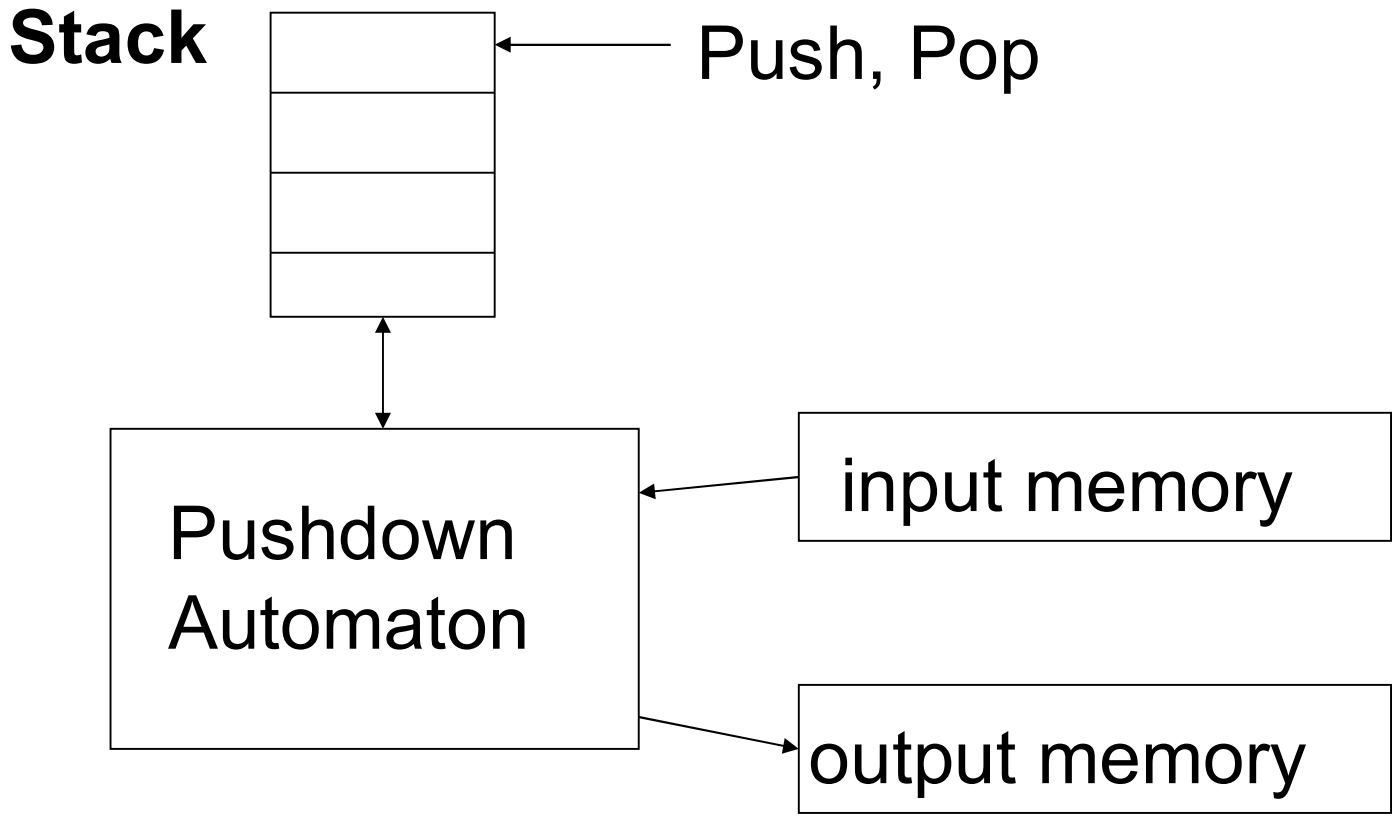
- **Finite Automata**: no temporary memory
- **Pushdown Automata**: stack
- **Turing Machines**: random access memory

Finite Automaton



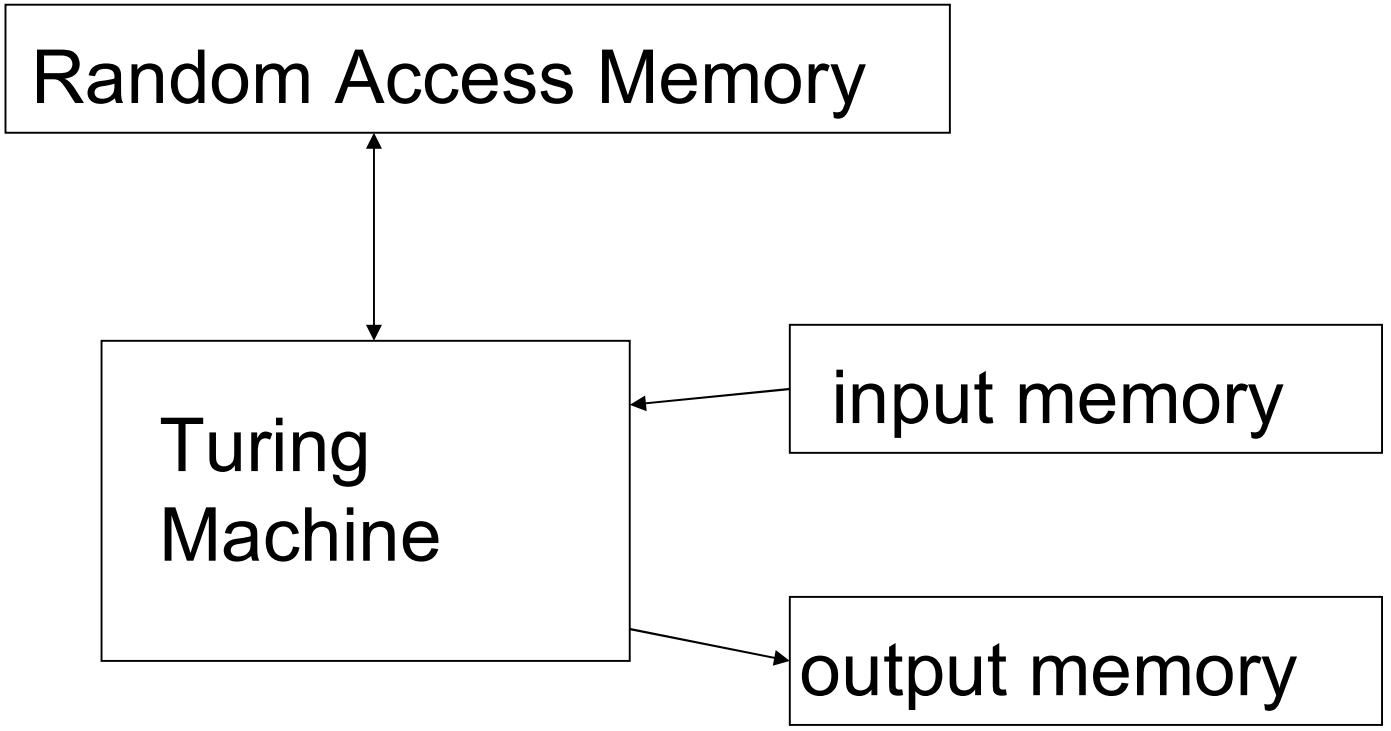
Example: Vending Machines
(small computing power)

Pushdown Automaton



Example: Compilers for Programming Languages
(medium computing power)

Turing Machine



Examples: Any Algorithm
(highest computing power)

Power of Automata

Less power ⟶ More power

Solve more computational problems

Overview of Computation

- Introduction to the Theory of Computation
- Finite Automata (A)
- Regular Languages and Grammars (A)
- Context-Free Languages (B)
- Pushdown Automata (B)
- Turing Machines (C)
- Limits of Algorithmic Computation
- Computational Complexity

- (A): What can we compute with only finite memory?
- (B): Computation with a stack memory
- (C): General computation and its inherent limitations

Introduction

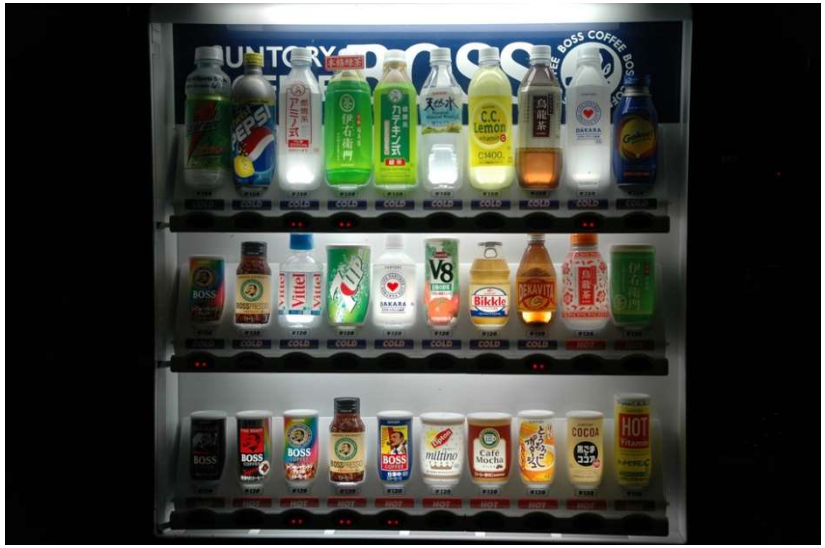
Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Formal Languages
  - Abstraction of the general characteristics of programming language
  - Consists of a set of symbols (string) and some rules (grammar) of formation by which these symbols can be combined into sentences

Introduction

Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Automata Theory
  - A question
    - Do you know how a vending machine works? Can you design one?



Introduction

Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Automata Theory
  - An example

    How to design a vending machine?

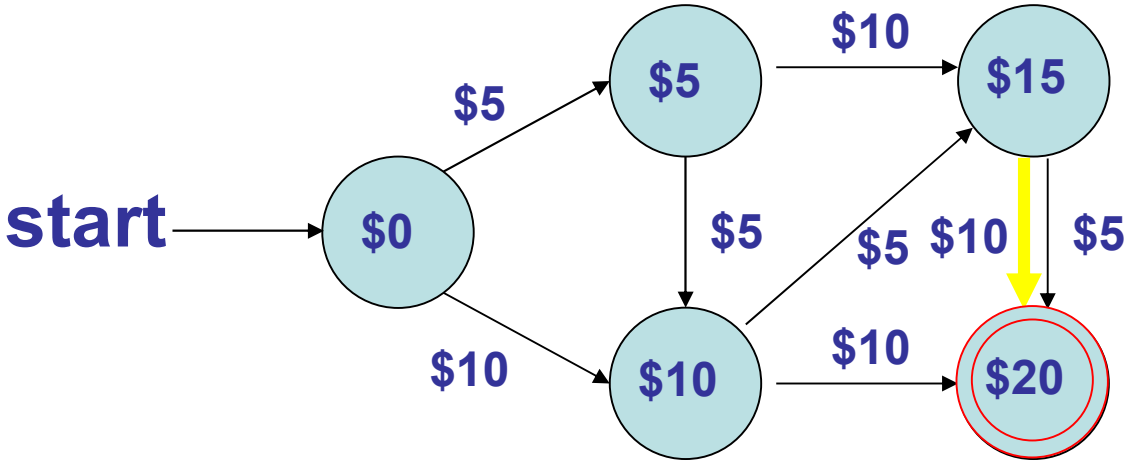
    □ *Use a finite automaton!*

  Assume (for simplicity):
  - Only NT 5-dollar and 10-dollar coins are used.
  - Only drinks all of 20 dollars are sold.

Introduction

Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

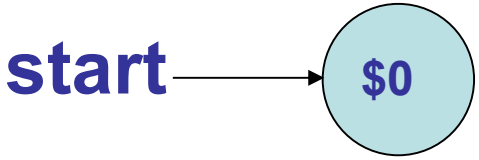
- Automata Theory
  - An example --- need “memory” called “states”



transition diagram

Notes:

$5 is returned as “output” (not shown)



“Start state”

“Final state”

Introduction

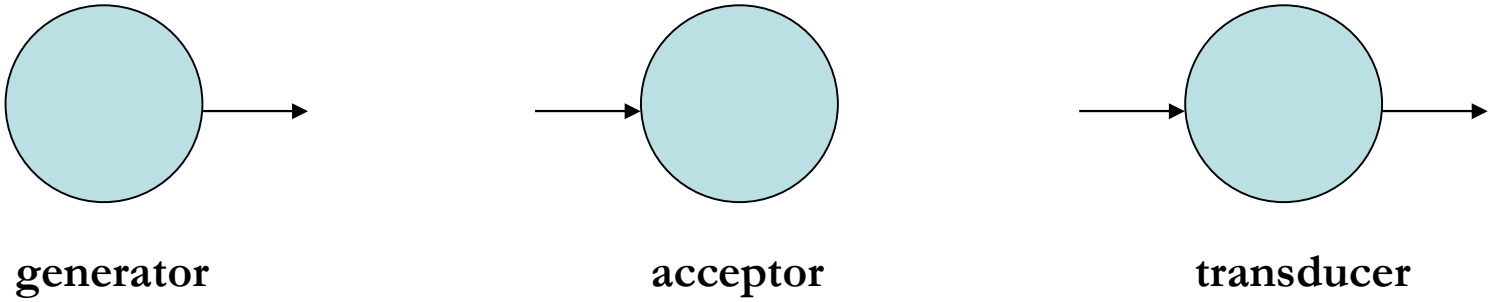
Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Automata Theory
  - Definition
    - study of dynamic behaviors of “discrete-parameter information systems” in form of “abstract computing devices, or “machines”
  - Examples of discrete-parameter information systems
    - digital systems, nerve systems, languages
    - information transmission systems
    - human-environment interactions, …

Introduction

Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Automata Theory
  - Three major models of automata
    - generator --- with output and without input
    - acceptor --- with input and without output
    - transducer --- both with input and with output



Introduction

Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Automata Theory
  - Examples
    - generator
      - "natural" grammar (generating "sentences" spoken by people)
      - Reception robot (speaking organized words and sentences)
      - context-free grammar (generating strings of symbols) *
    - (* abstract models studied in this course)

Reception robot
--- Expo 2005

Introduction

Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Automata Theory
  - Examples
    - acceptor
      - digital lock (accepting digits)
      - lexical analyzer (recognizing keywords in computer languages)
      - finite automaton (accepting valid strings of symbols) *
    - (* abstract models studied in this course)

Digital lock

Introduction

Theory of computation:
- Formal languages
- Automata theory
- Computability
- Complexity

- Automata Theory
  - Examples
    - transducer
      - Interpreter (translating natural languages)
      - Compiler (translating high-level languages into machine codes)
      - Turing machine (transforming strings of symbols) *
    - (* abstract models studied in this course)

"Interpreter"
(a movie)

Introduction

Theory of computation:
Formal languages
Automata theory
Computability
Complexity

- Computability
  - Definition
    - study of problem solving capabilities of computational models
  - Problem types based on resources
    - Impossible problems
    - Possible-with-unlimited-resources-but-impossible-with-limited-resources problems
    - Possible-with-limited-resources problems

Introduction

Theory of computation:
Formal languages
Automata theory
Computability
Complexity

- Computability
  - Problem types based on time
    - Undecidable problems
    - Intractable problems
    - Tractable problems
  - Studies of computability help us not to waste time on “unsolvable problems” already investigated before.

Introduction

Theory of computation:
Formal languages
Automata theory
Computability
Complexity

- Computational complexity
  - Definition
    - study of "efficiency" of problem solving.
  - To unify comparison, an abstract model is needed as the machine for executing problem solutions.
  - Usually the most famous "Turing machine" (an automaton) is used.

Introduction

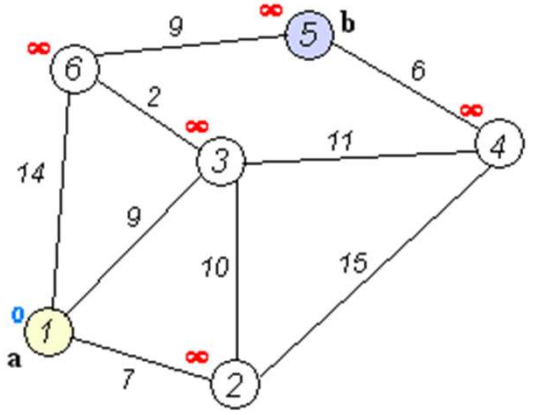
Theory of computation:
Formal languages
Automata theory
Computability
Complexity

- Computational complexity
  - Turing machine, though simple, has been proved to be able to simulate any problem solving steps ("algorithms") designed by human beings!
  - Turing machine is the foundation of modern computation theory development!

The Shortest Path Problem

P (Polynomial)

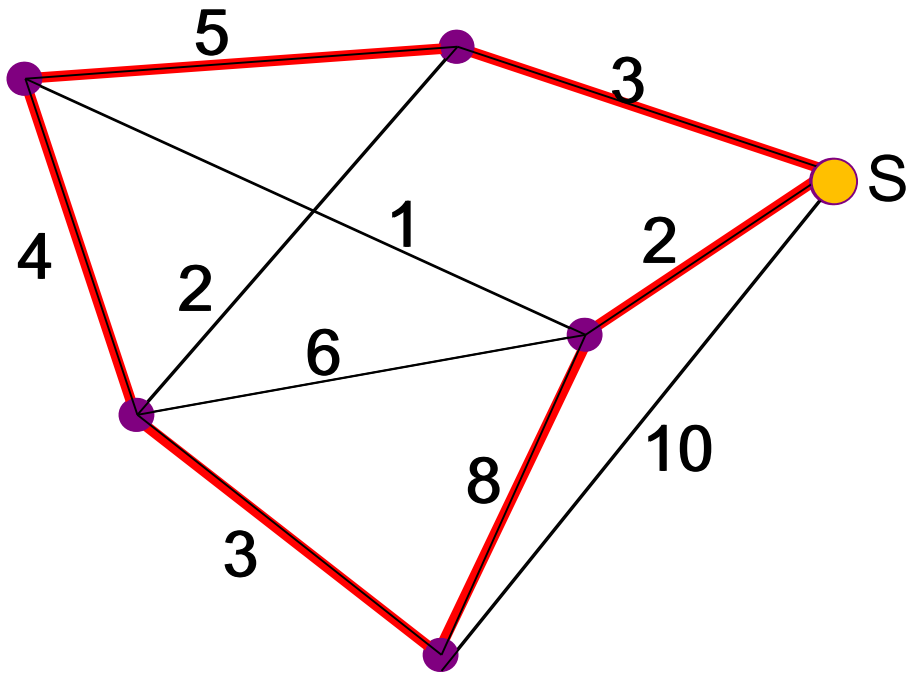
- Given:
  - Directed graph $G = (V, E)$
    - Length l_e = length of edge $e = (u, v) \in E$
      - Distance; time; cost
      - $l_e \geq 0$
  - Source s
- Goal:
  - Shortest path P_v from s to each other node $v \in V - \{s\}$
    - Length of path P: $l(P) = \Sigma_{e \in E} l_e$



$l(a \rightarrow b) = l(1 \rightarrow 3 \rightarrow 6 \rightarrow 5)$
$= 9+2+9 = 20$

Basic: $O(|V|^2)$
Fibonacci Heap: $O(|E| + |V| \log |V|)$

Example: the Traveling Salesman Problem



Cost: 25

What is the least-cost round-trip route that visits each city exactly once and then returns to the starting city?

Brief History of Theory of Computation

- In 1936, Turing proposed the model of universal algorithm machine, which later was called Turing machine.
- Some similar ideas were proposed by other scholars, including Stephen C. Kleene, Alonzo Church, Emil Post, etc., in different forms of models.

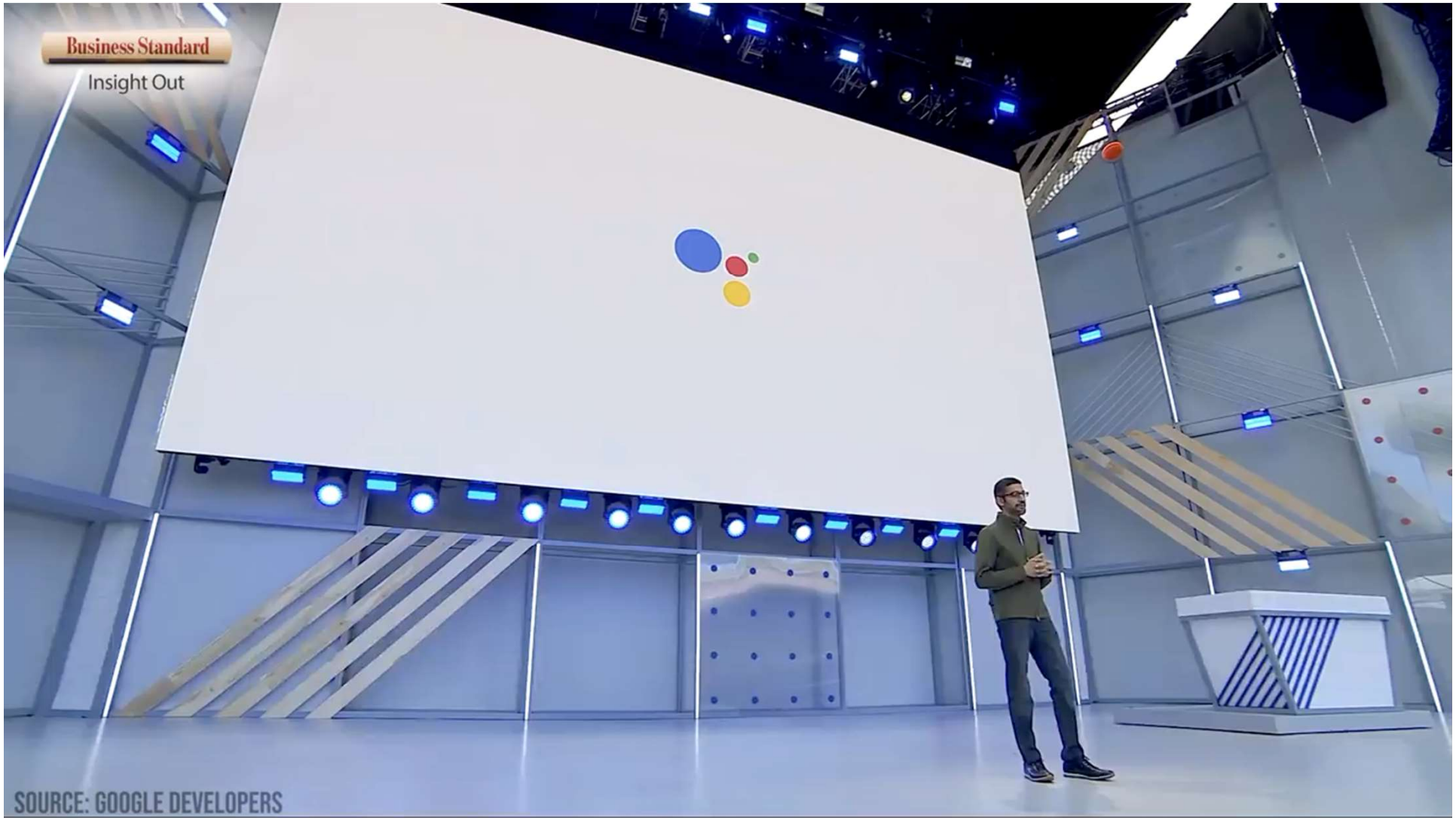
University of Cambridge (N/A)

照片來源： flickr

> 圖靈測試（英語：Turing test，又譯圖靈試驗）是圖靈於1950年提出的一個關於判斷機器是否能夠思考的著名試驗，測試某機器是否能表現出與人等價或無法區分的智力。如果一個人（代號C）使用測試對象皆理解的語言去詢問兩個他不能看見的對象任意一串問題。對象為：一個是正常思維的人（代號B）、一個是機器（代號A）。如果經過若干詢問以後，C不能得出實質的區別來分辨A與B的不同，則此機器A通過圖靈測試。 - 維基百科

模仿遊戲 2015

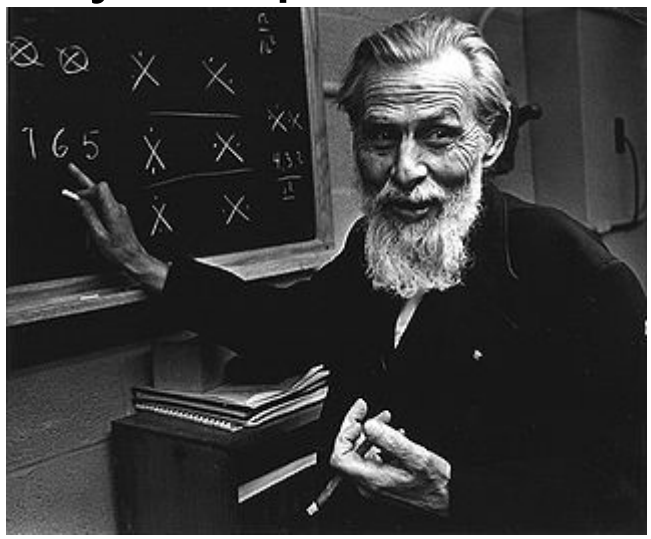
Holy grail



Brief History of Theory of Computation

- In 1943, neural physiologists Warren S. McCulloch and Walter Pitts developed finite-state systems to simulate neural networks in biological systems.

- They are pioneers of automata theory.



University of Chicago (N/A)

MIT(N/A)

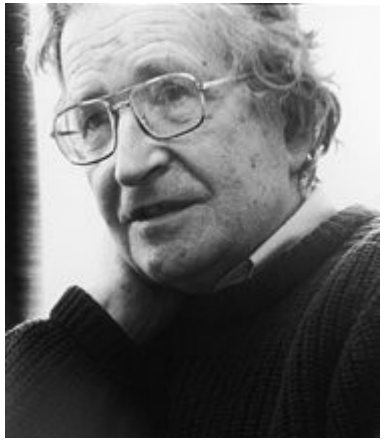
Brief History of Theory of Computation

- In late 1940's, Von Neumann proposed the idea of stored program for computer models.
- In 1951, a real computer following this idea was constructed (UNIVAC I, the world's first commercially available computer, by Eckert-Mauchly Computer Company).

Princeton University (N/A)

Brief History of Theory of Computation

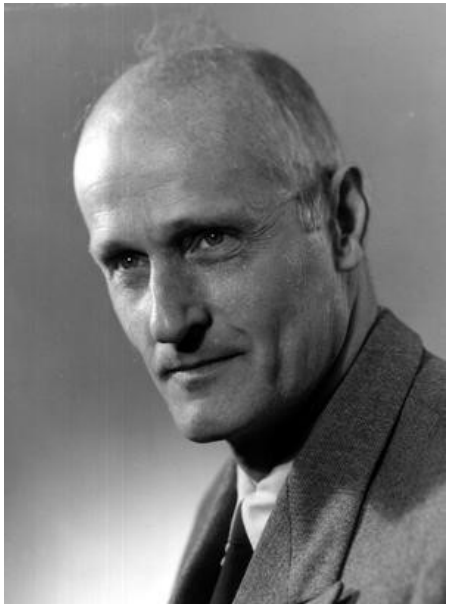
- In late 1950's, linguist N. Chomsky proposed a mathematical model for grammars of natural languages.
- In 1956, he propose further the concept of context-free grammar, which may be used for defining computer languages.



Retired @ MIT

Brief History of Theory of Computation

- In 1956, Kleene proposed the concept of finite automaton for simulating the neural network proposed by McCulloch and Pitts.
- He also proposed regular expressions to describe strings of symbols, and proved them equivalent to finite automata.



University of Wisconsin-Madison (N/A)

Brief History of Theory of Computation

- In 1959 and 1960, John W. Backus & Peter Naur proposed sequentially a special expression for grammars of computer languages, called later Backus-Naur form (BNF).
  - It may be used to describe the computer language ALGOL-60, leading to intensive development of compiling theory.

IBM (N/A)

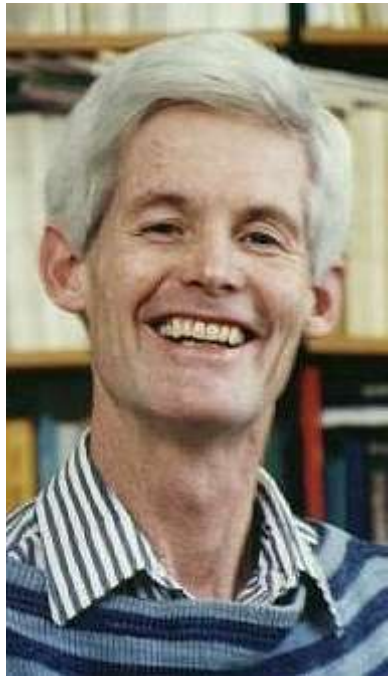
•Turing Award

Technical University of Denmark
Copenhagen University

•Turing Award

Brief History of Theory of Computation

- In 1969, Stephen A. Cook found the problems can be separated into tractable and intractable ones.
  - Intractable problems are also called NP-hard problems.
  - Such problems cannot be solved by computers except very small instances (with only small-sized inputs).



University of California, Berkeley

•Turing Award

Question?