CSE3064

Formal Languages and Automata Theory

Original Slides by C. Busch, LSU.

General Info about Course

Text Book:

- Introduction to the Theory of Computation, Michael Sipser, 3rd Edition, Cengage Learning.

Reference Book:

- J. E. Hopcroft, R. Motwani, J. D. Ullman, "Introduction to Automata Theory, Languages, and Computation", 3rd Edition, Pearson.

Grading:

- Midterm 30%, Homework 30%, Final 40%.
- 70% attendance is required!

Course Goals

Provide computation models

Analyze power of models

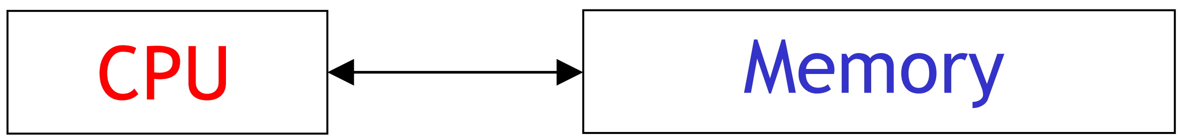
Answer intractability questions:

What computational problems can each model solve?

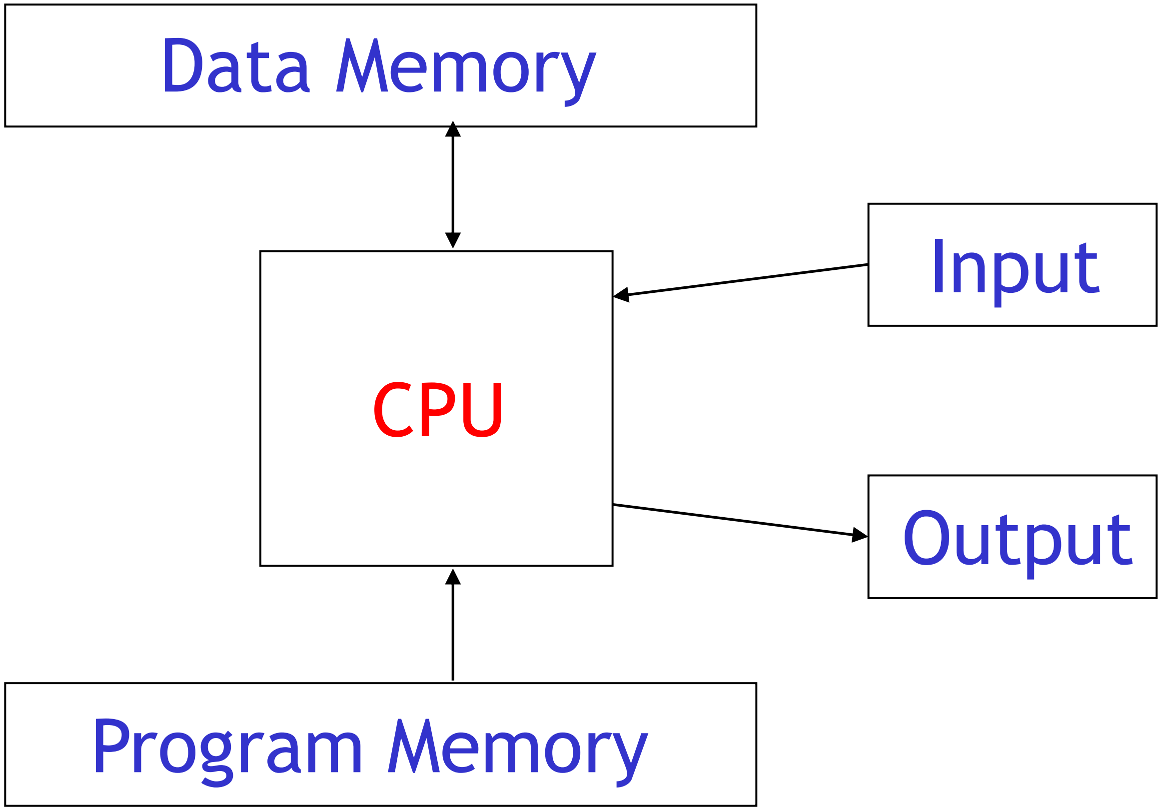
Answer time complexity questions:

How much time we need to solve the problems?

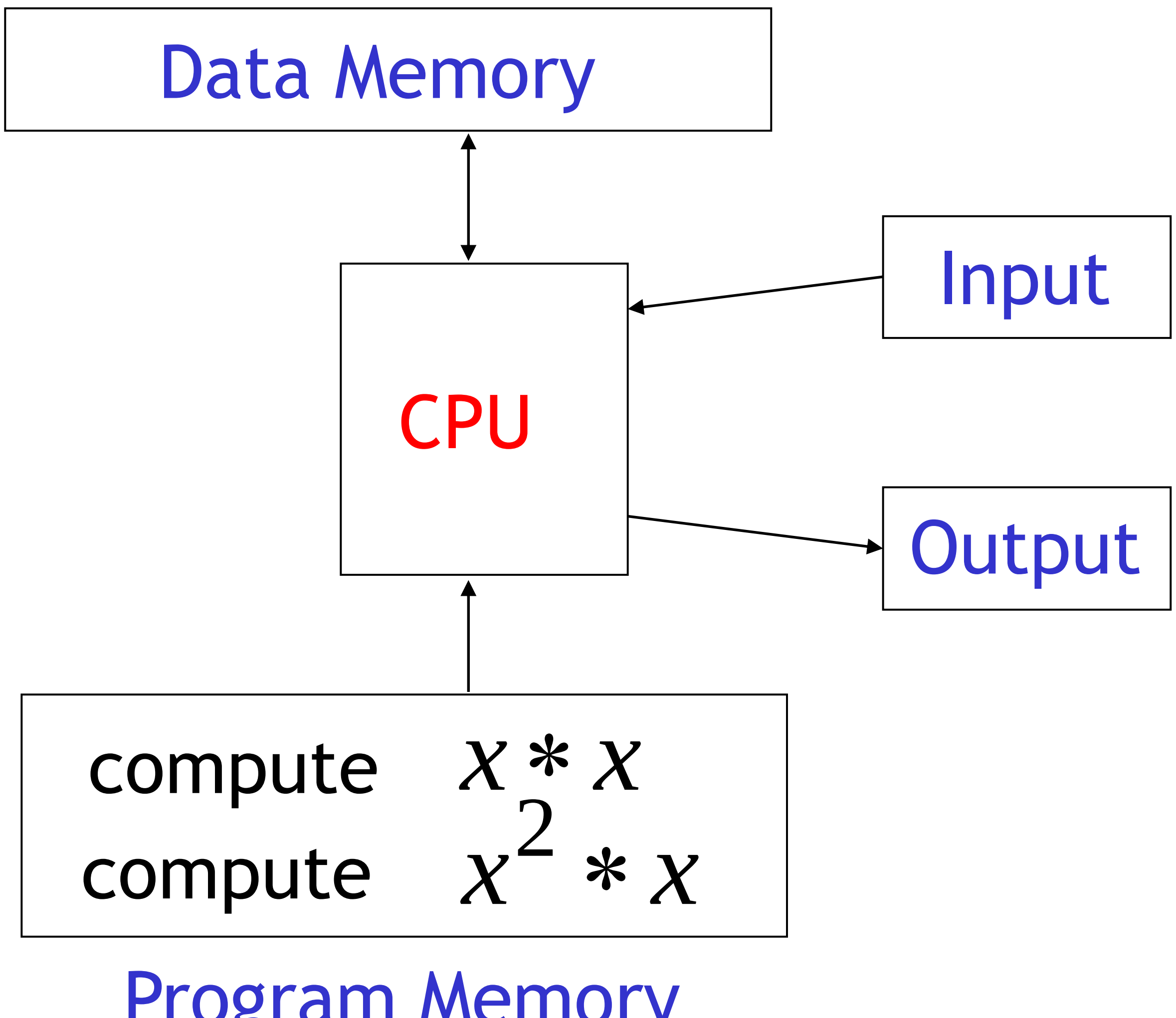
A widely accepted model of computation



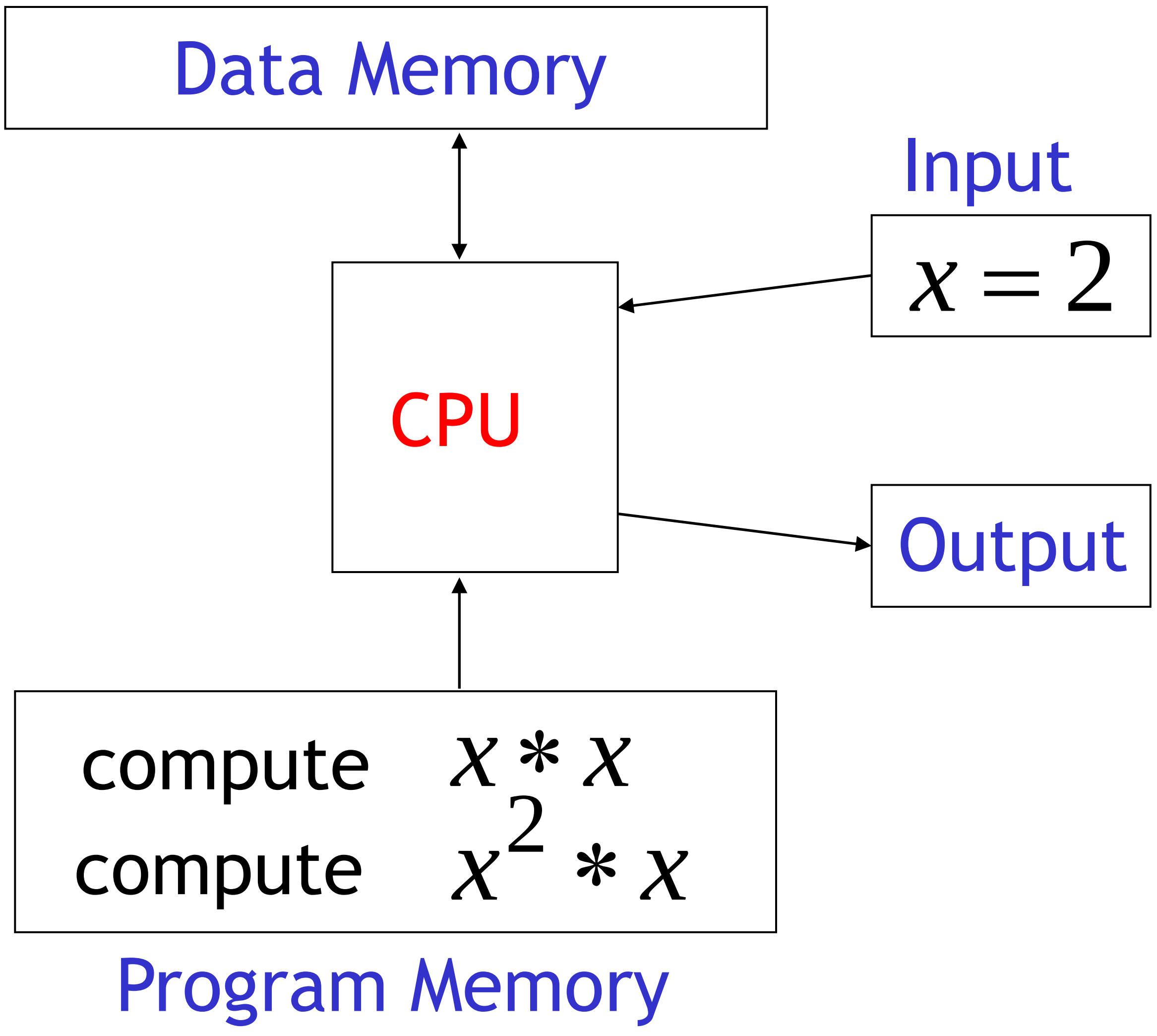
The different components of memory



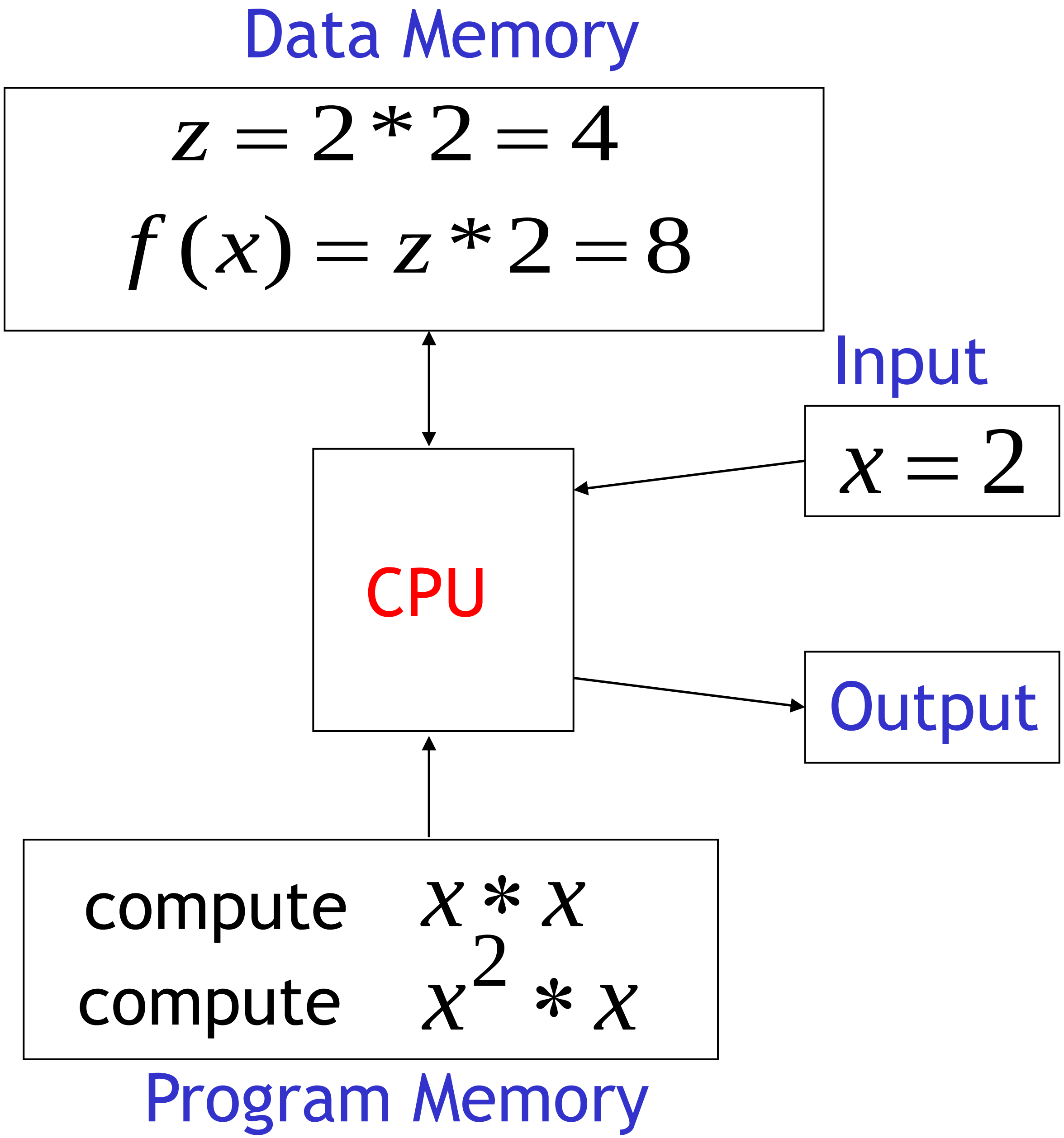
Example: $f(x) = x^3$



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

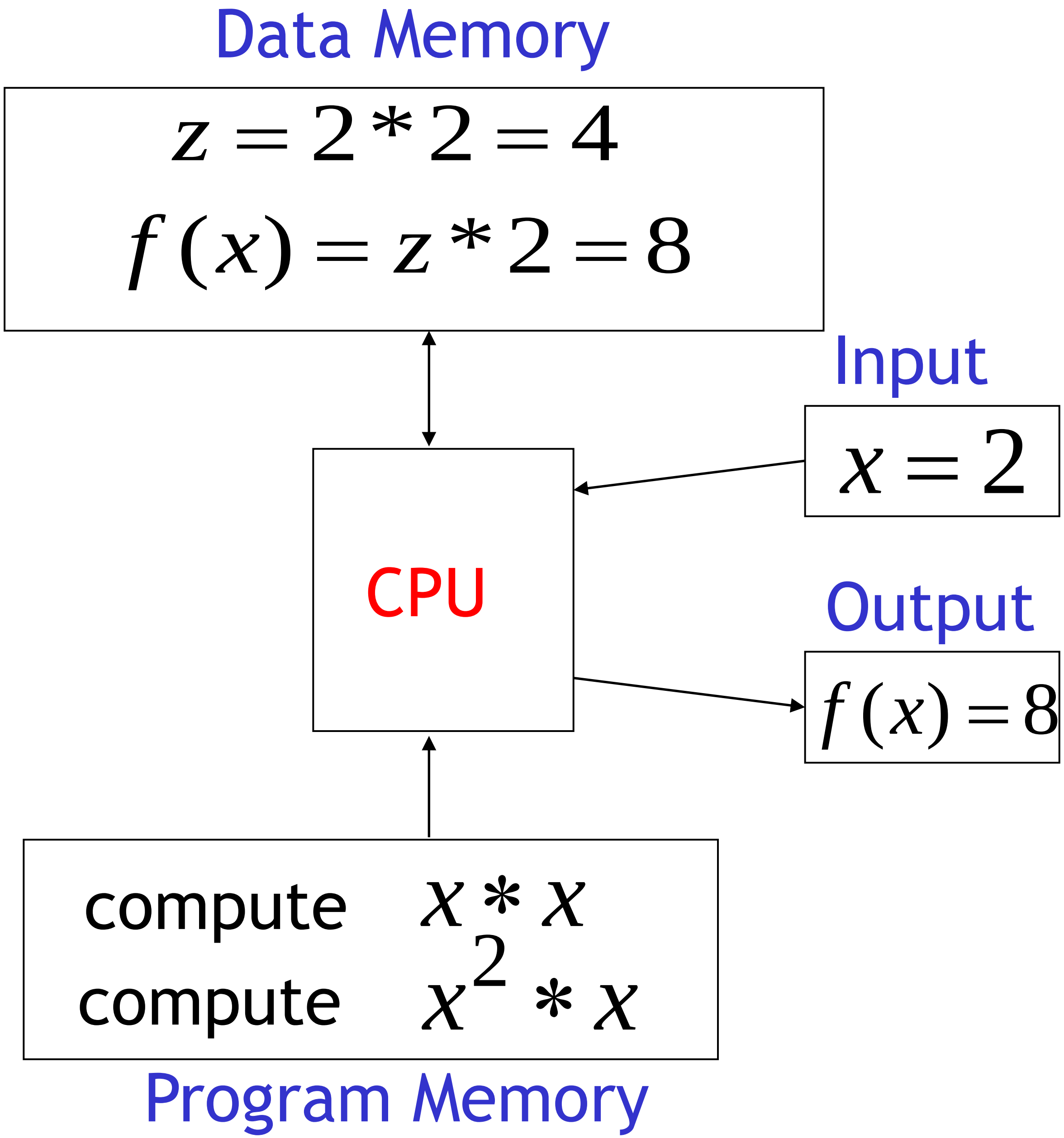


$$f(x) = x^3$$



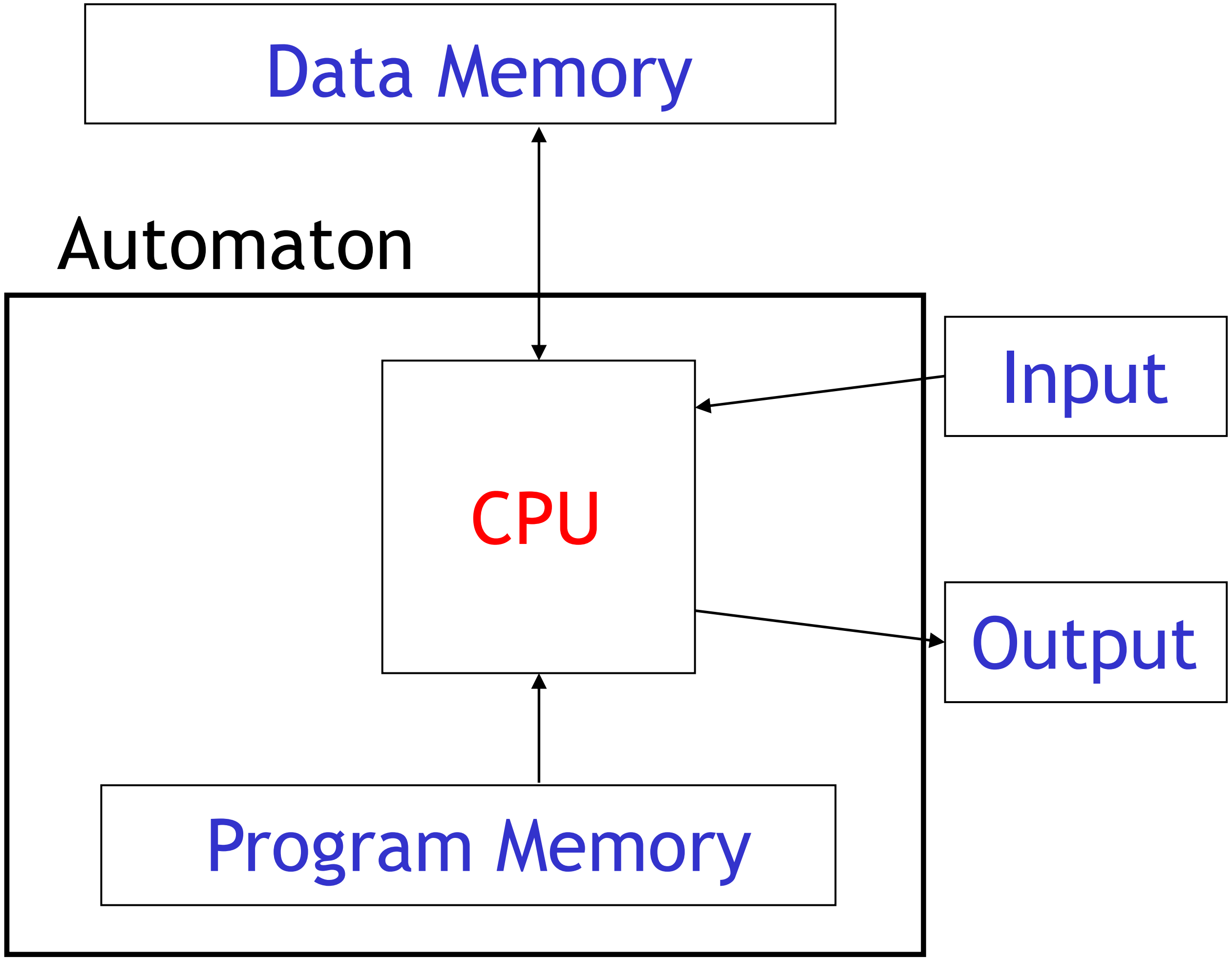
Data Memory
$z = 2 * 2 = 4$
$f(x) = z * 2 = 8$
Input
$x = 2$
CPU
Output
compute $x * x$
compute $x^2 * x$
Program Memory

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

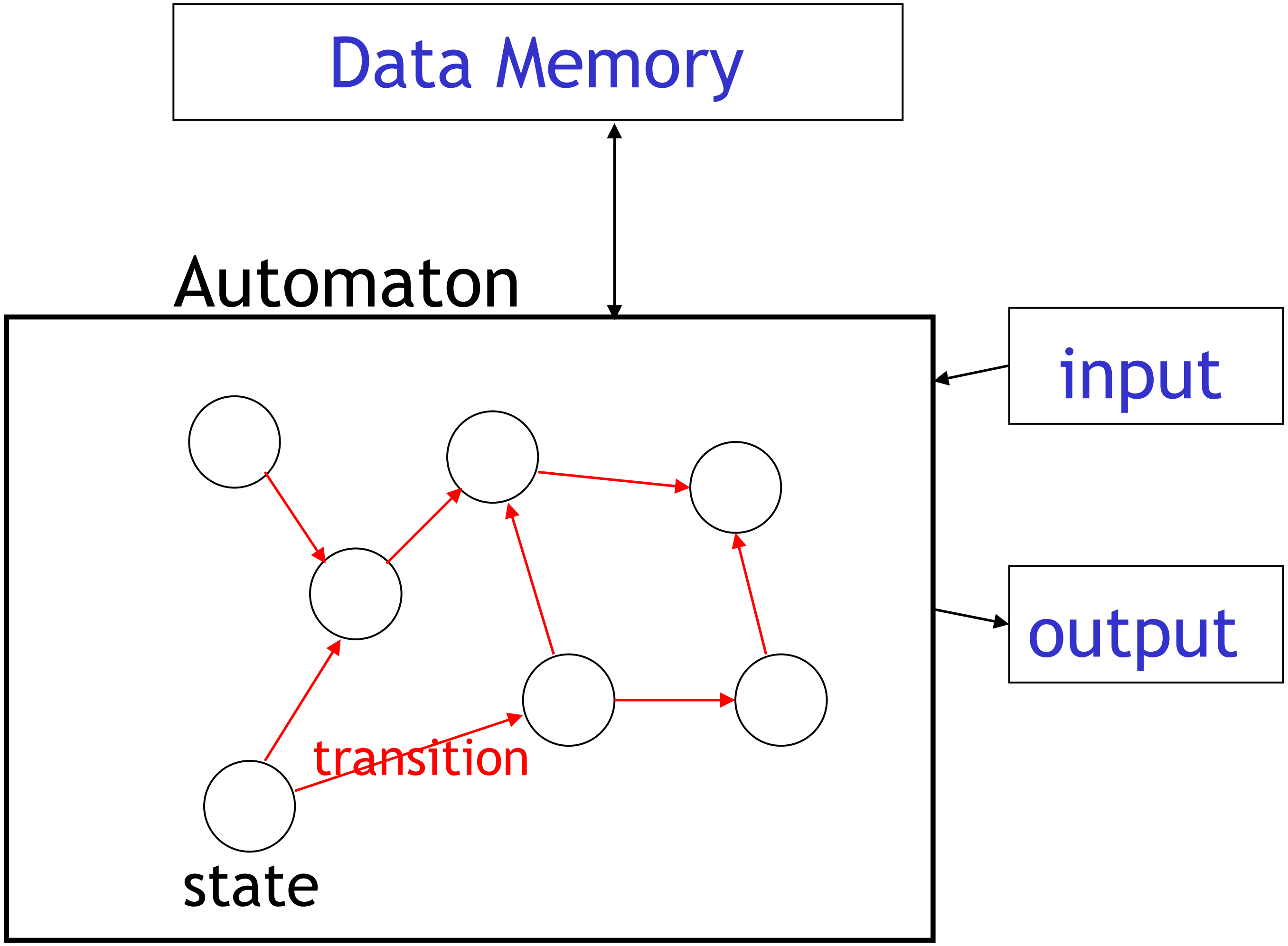


Data Memory
$z = 2 * 2 = 4$
$f(x) = z * 2 = 8$
Input
$x = 2$
CPU
Output
$f(x) = 8$
compute $x * x$
compute $x^2 * x$
Program Memory

Automaton



Automaton



CPU + Program Memory = States + Transitions

Different Kinds of Automata

Automata are distinguished by the data memory

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

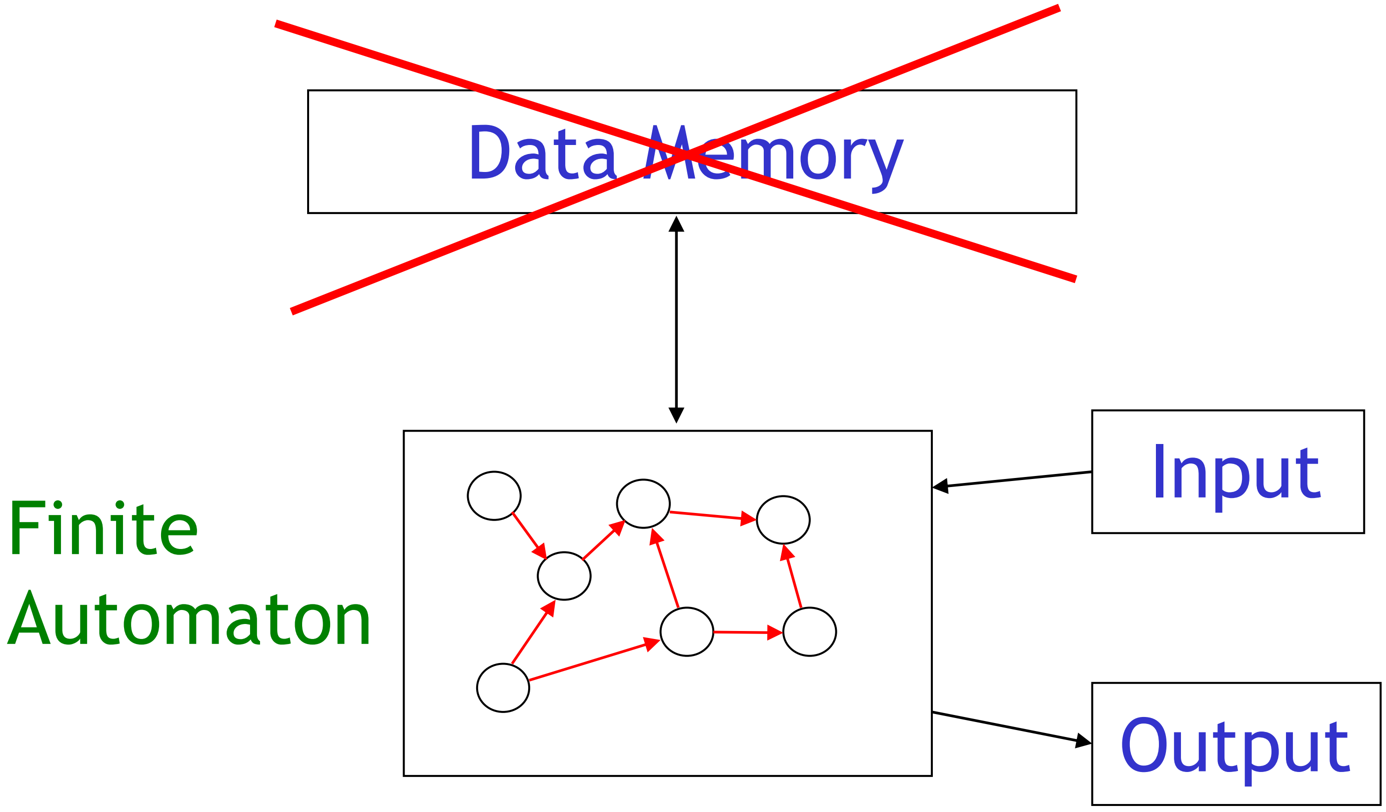
Memory affects computational power:

More flexible memory

results to

the solution of more computational problems

Finite Automaton

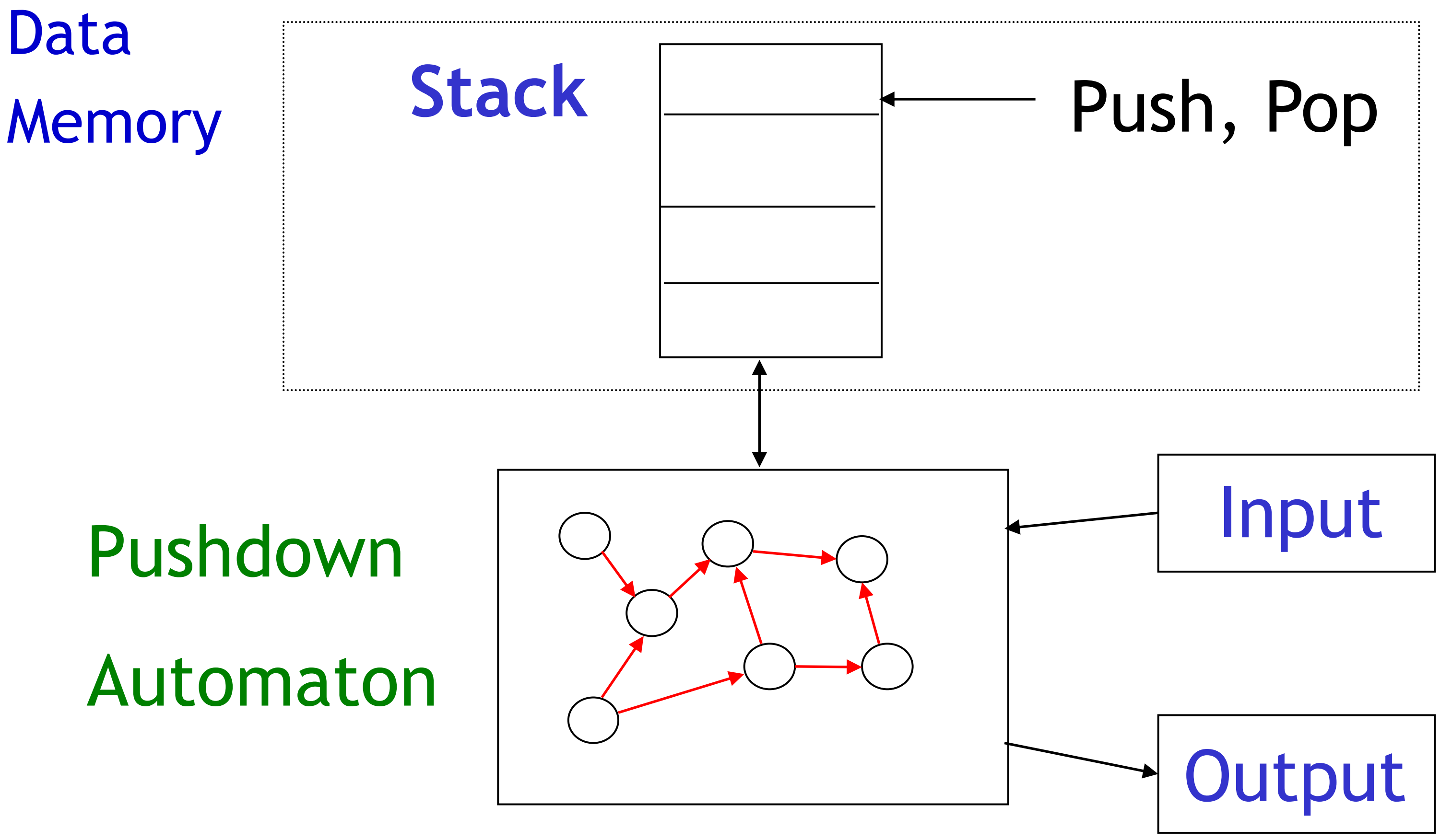


Example: Elevators, Washing Machines, Tea Maker,

Lexical Analyzers

(Small Computing Power)

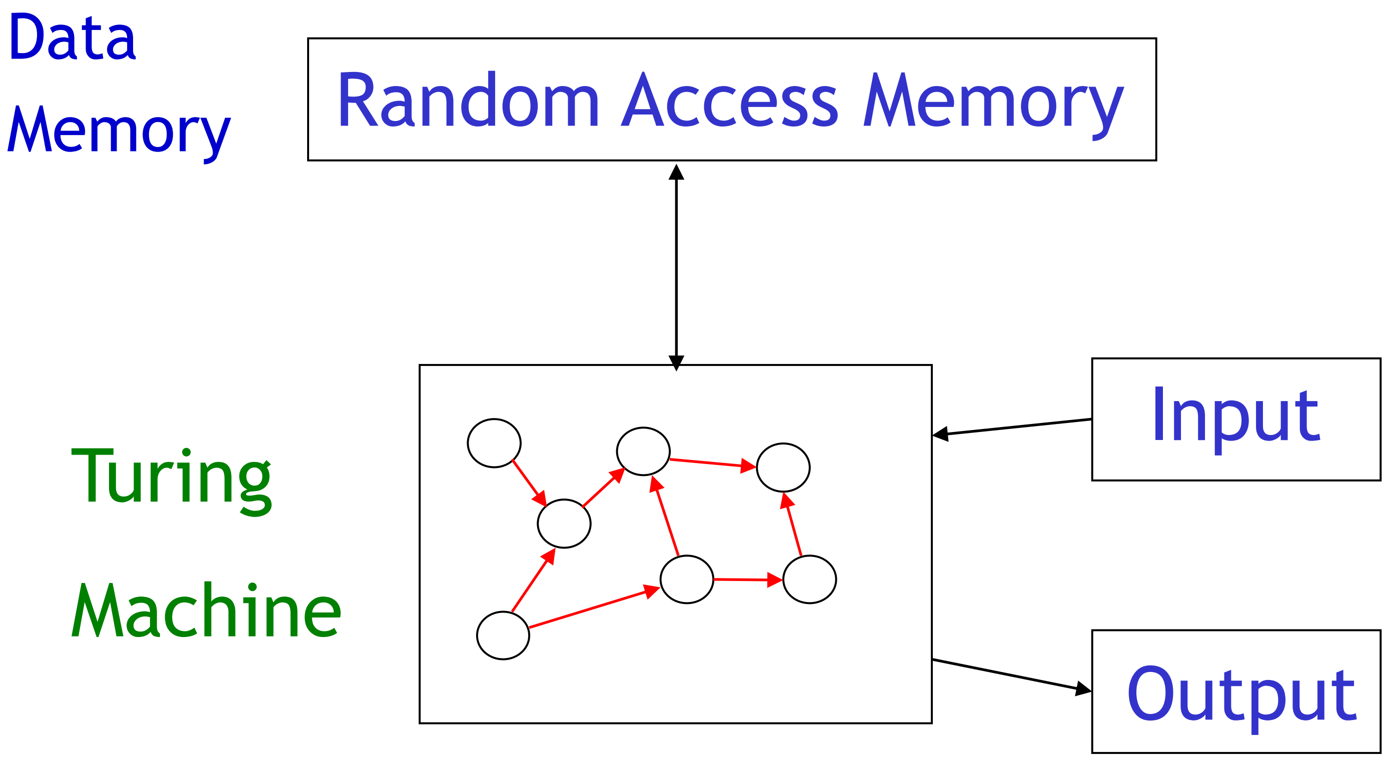
Pushdown Automaton



Example: Parsers for Programming Languages

(Medium Computing Power)

Turing Machine



Examples: Any Algorithm

(Highest Known Computing Power)

Power of Automata

Simple problems		More complex problems		Hardest problems
Finite Automata	<	Pushdown Automata	<	Turing Machine

Less power → More power

Solve more computational problems

Turing Machine is the most powerful known computational model

Question: Can Turing Machines solve all computational problems?

Answer: No!

(There are unsolvable problems)

Time Complexity of Computational Problems:

P problems:

(Polynomial time problems)

Solved in polynomial time

NP-complete problems:

(Non-deterministic Polynomial time problems)

Believed to take exponential time to be solved