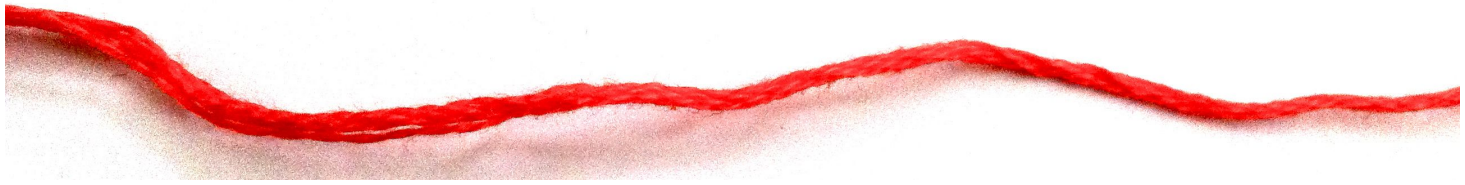
Automata & Complexity :: Introduction

Jörg Endrullis

Vrije Universiteit Amsterdam

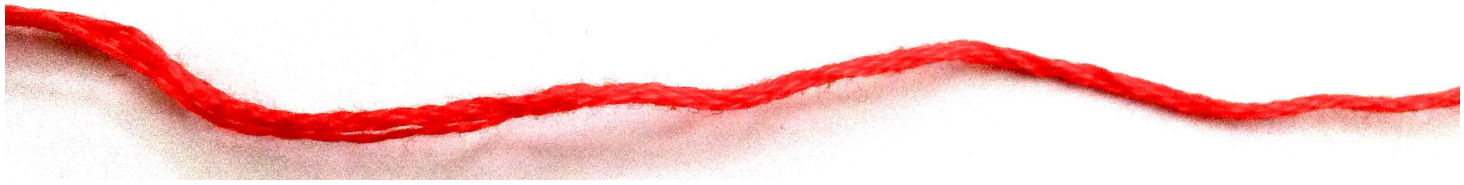
Motivation

Computer is based on a **universal computation mechanism**.



Motivation

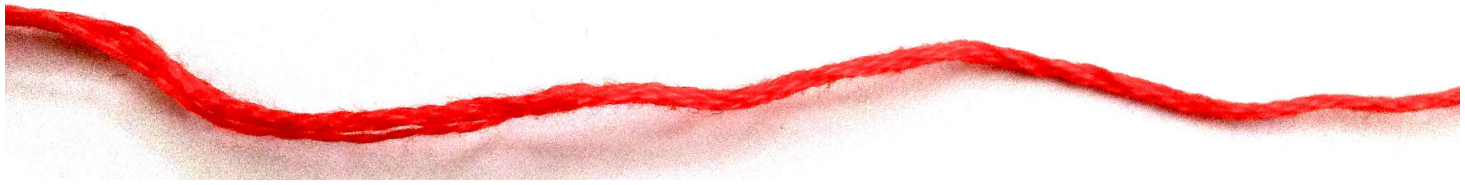
Computer is based on a **universal computation mechanism**.



What does **universal** mean?

Motivation

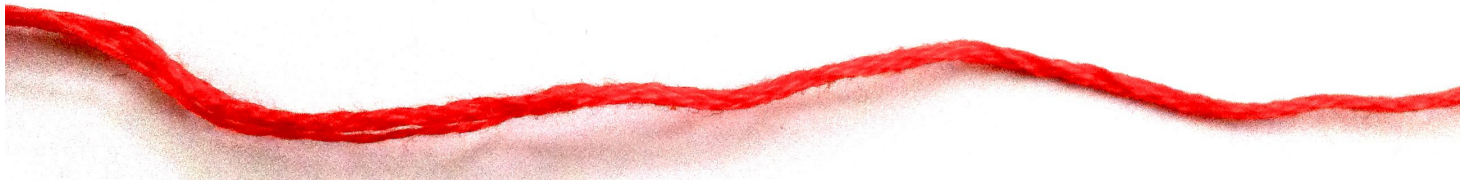
Computer is based on a **universal computation mechanism**.



What does **universal** mean? Roughly speaking, no other computation model can compute more.

Motivation

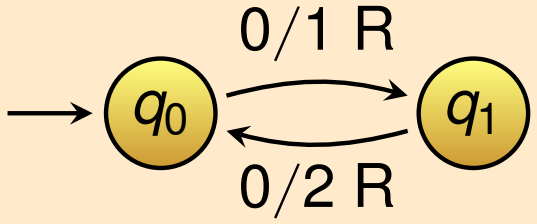
Computer is based on a **universal computation mechanism**.



What does **universal** mean? Roughly speaking, no other computation model can compute more.

Turing machines were invented in 1936 by Alan Turing:

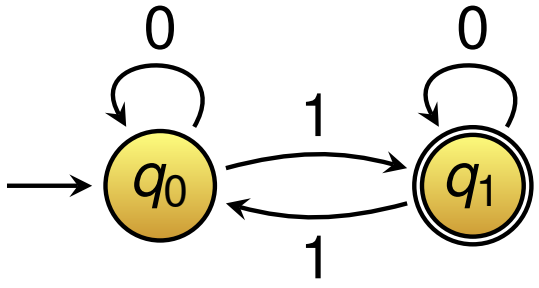
- abstract computation model
- **automata** with read/write tape
- universal computation mechanism



Universal computation invented before the first computers!

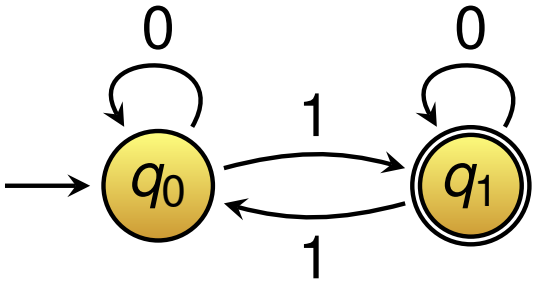
Motivation

Different kinds of automata for different applications.



Motivation

Different kinds of automata for different applications.

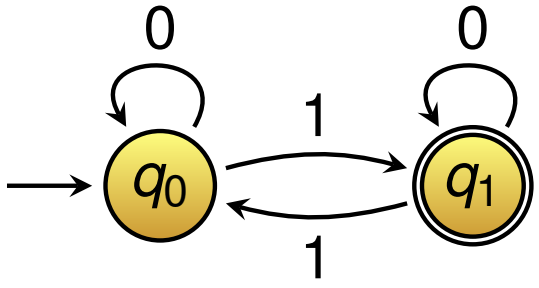


Finite automata give rise to regular languages:

- application: **pattern recognition**
- equivalent to: **regular expressions**, regular grammars

Motivation

Different kinds of automata for different applications.



Finite automata give rise to regular languages:

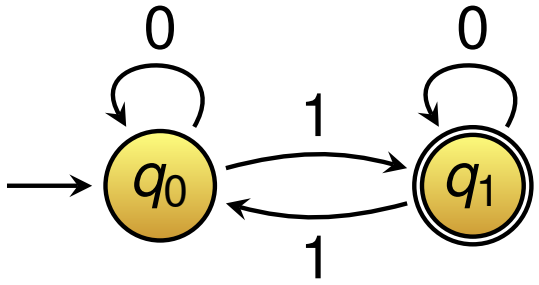
- application: **pattern recognition**
- equivalent to: **regular expressions**, regular grammars

Pushdown automata give rise to context-free languages:

- application: **parsing** (e.g. programming languages)
- equivalent to context-free grammars

Motivation

Different kinds of automata for different applications.



Finite automata give rise to regular languages:

- application: **pattern recognition**
- equivalent to: **regular expressions**, regular grammars

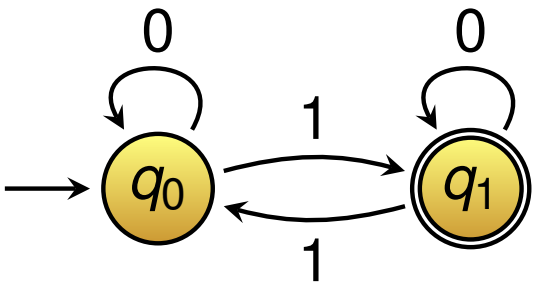
Pushdown automata give rise to context-free languages:

- application: **parsing** (e.g. programming languages)
- equivalent to context-free grammars

Turing machines yield recursively enumerable languages:

- application: general **computation**

Automata are Ubiquitous



- Number theory
- Formal verification
- Compiler construction
- Software modelling
- Hardware design
- Parsing
- Pattern matching
- Theory of computation

Central Question: What can a Computer do?

What can a computer do?

Central Question: What can a Computer do?

What can a computer do?

Some (at first glance simple) problems are undecidable.

For example:

- program termination
- Post correspondence problem
- validity in predicate logic

Central Question: What can a Computer do?

What can a computer do?

Some (at first glance simple) problems are undecidable.

For example:

- program termination
- Post correspondence problem
- validity in predicate logic

Some problems (NP-hard problems) are (probably) not efficiently solvable by a computer.

For example:

- travelling salesman problem
- satisfiability in propositional logic

Typical Questions

- What is a (programming) **language**?
- How can languages be recognised by computers?
  - **automata**
  - **grammars**
  - **regular expressions**
- What problems can be solved by what **types of automata**?
- **How much time/memory** is needed for solving a problem?

Typical Questions

- What is a (programming) **language**?
- How can languages be recognised by computers?
  - **automata**
  - **grammars**
  - **regular expressions**
- What problems can be solved by what **types of automata**?
- **How much time/memory** is needed for solving a problem?

Aspects of languages:

- **syntax**: the form of the words in the language
- **semantics**: the meaning of the words in the language

We will focus on the syntax.