CS 133 : Automata Theory and Computability

Lecture Slides

An Introduction

Francis George C. Cabarle

Algorithms and Complexity Laboratory
Department of Computer Science
University of the Philippines Diliman
fccabarle@up.edu.ph

Day 1

An Introduction

Automata Theory

Foundations of Automata Theory

A Prelude

Introduction

Theory of Computation

- Fundamental mathematical properties of computer hardware, software, and applications
- Determine what can and cannot be computed, how quickly, with how much memory, and on which type of computational model

Theory of Computation

Central Areas:

[1] and Formal language theory, but to a lesser degree in this course.

Theory of Computation

Central Areas:

1. Automata theory[1],

[1] and Formal language theory, but to a lesser degree in this course.

Theory of Computation

Central Areas:

1. Automata theory[1],
2. Computability,

[1] and Formal language theory, but to a lesser degree in this course.

Theory of Computation

Central Areas:

1. Automata theory[1],
2. Computability,
3. Complexity.

[1] and Formal language theory, but to a lesser degree in this course.

Theory of Computation

Central Areas:

1. Automata theory[1],
2. Computability,
3. Complexity.

$\Longrightarrow$ What are the fundamental capabilities and limitations of computers?

[1] and Formal language theory, but to a lesser degree in this course.

Theory of Computation

Central Areas:

1. Automata theory[1],
2. Computability,
3. Complexity.

Classification of problems as easy ones and hard ones

[1] and Formal language theory, but to a lesser degree in this course.

Theory of Computation

Central Areas:

1. Automata theory[1],
2. Computability,
3. Complexity.

Classification of problems by those that are solvable and those that are not solvable

[1] and Formal language theory, but to a lesser degree in this course.

Theory of Computation

Central Areas:

1. Automata theory[1],
2. Computability,
3. Complexity.

Deals with the definitions and properties of mathematical models of computation

[1] and Formal language theory, but to a lesser degree in this course.

What is Automata Theory?

Automata theory is the study of abstract computing devices and their capacities.

What is Automata Theory?

Automata theory is the study of abstract computing devices and their capacities.

- an excellent place to begin the study of the theory of computation

What is Automata Theory?

Automata theory is the study of abstract computing devices and their capacities.

- an excellent place to begin the study of the theory of computation
- allows practice with formal definitions of computation

Why Study Automata Theory?

[2] http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html

Why Study Automata Theory?

- IT IS REQUIRED FOR YOUR BSc DEGREE.

[2] http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html

Why Study Automata Theory?

- IT IS REQUIRED FOR YOUR BSc DEGREE.
- It gives you certain models for computation.

[2] http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html

Why Study Automata Theory?

- IT IS REQUIRED FOR YOUR BSc DEGREE.
- It gives you certain models for computation.
- It gives you a notion of what a computer can and cannot do.

[2] http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html

Why Study Automata Theory?

- IT IS REQUIRED FOR YOUR BSc DEGREE.
- It gives you certain models for computation.
- It gives you a notion of what a computer can and cannot do.
- FOR FUN!!! e.g.

[2] http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html

Why Study Automata Theory?

- IT IS REQUIRED FOR YOUR BSc DEGREE.
- It gives you certain models for computation.
- It gives you a notion of what a computer can and cannot do.
- FOR FUN!!! e.g.
  - Alan Turing, The Enigma (book), The Imitation Game (film)

[2] http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html

Why Study Automata Theory?

- IT IS REQUIRED FOR YOUR BSc DEGREE.
- It gives you certain models for computation.
- It gives you a notion of what a computer can and cannot do.
- FOR FUN!!! e.g.
  - Alan Turing, The Enigma (book), The Imitation Game (film)
  - Turing-complete games, tools, e.g. Pokémon Yellow, Magic: The Gathering, MediaWiki templates etc.[2]

[2] http://beza1e1.tuxen.de/articles/accidentally_turing_complete.html

Some Applications of Automata Theory

Some Applications of Automata Theory

- (String) Pattern matching
  $\Longrightarrow$ Regular Expressions

Some Applications of Automata Theory

- (String) Pattern matching
  $\Longrightarrow$ Regular Expressions

- Programming languages, compilers
  $\Longrightarrow$ Context-Free Languages / Grammars

Some Applications of Automata Theory

- (String) Pattern matching
  $\Longrightarrow$ Regular Expressions

- Programming languages, compilers
  $\Longrightarrow$ Context-Free Languages / Grammars

- Modeling and simulation e.g. Petri nets, Markov chains, finite state machines (transducers), cellular automata
  $\Longrightarrow$ variants of finite automata!

Introduction

Automata Theory

Foundations of Automata Theory

A Prelude

Discrete Mathematics

- Sets
- Sequences and Tuples
- Functions and Relations
- Graphs
- Boolean Logic
- Proving Techniques

Linguistics

- Alphabets
- Strings
- Grammar
- Languages

Introduction

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

- What is an example of an input?

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

- What is an example of an input?

Solution: A string of inputs (actions) *beginning* at the initial state and *ending* at the final state

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

- What is an example of an input?

Solution: A string of inputs (actions) *beginning* at the initial state and *ending* at the final state

- What is a "short" solution? Length(s) of such solutions?

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

- What is an example of an input?

Solution: A string of inputs (actions) *beginning* at the initial state and *ending* at the final state

- What is a "short" solution? Length(s) of such solutions?
- What is the state transition diagram?

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

- What is an example of an input?

Solution: A string of inputs (actions) *beginning* at the initial state and *ending* at the final state

- What is a "short" solution? Length(s) of such solutions?
- What is the state transition diagram?
- What is the state transition table?

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

- What is an example of an input?

Solution: A string of inputs (actions) *beginning* at the initial state and *ending* at the final state

- What is a "short" solution? Length(s) of such solutions?
- What is the state transition diagram?
- What is the state transition table?
- Are there (in)finite solutions to the problem?

A finite (game) state system of survival!

Players: man (M), wolf (W), goat (G), cabbage (C).
Setup: left shore - river - right shore
Must not allow the following: G eats C, and W eats G.
(Memory) states: player(s) on left shore - player(s) on right shore, e.g. MGC-W (Okay), MW-GC (Not okay! Why?)

- What is the initial state? Final state?

(Actions) Inputs: man crosses the river alone (m), with goat (g), or wolf (w), or cabbage (c).

- What is an example of an input?

Solution: A string of inputs (actions) *beginning* at the initial state and *ending* at the final state

- What is a "short" solution? Length(s) of such solutions?
- What is the state transition diagram?
- What is the state transition table?
- Are there (in)finite solutions to the problem?
- Is the system a typical or atypical for finite state systems? How so?

Fin (wakas)

Thanks for the attention.

Questions?

Reading assignment(s)

- Chapter 0 of [Sipser 2005].

References:

[Sipser 2005] M. Sipser. *Introduction to the Theory of Computation*: 2ed. PWS Publishing Company, 2005.

[Hopcroft, Ullman 1979] J. Hopcroft, J. Ullman. *Introduction to Automata Theory, Languages, and Computation*: Addison-Wesley, 1979.

[Hernandez 2014] CS133 lecture slides of N.H.S. Hernandez, 2014