Automata theory and its applications
Lecture 1: Historical perspective, course syllabus, basic concepts

Zhilin Wu

State Key Laboratory of Computer Science,
Institute of Software, Chinese Academy of Sciences

September 26, 2012
Outline

1. What is automata theory
2. Why to bother with automata theory?
3. Historical perspective of automata theory
4. About this Course
5. Basic concepts
The origin of the word “automata”:

The Greek word “aутóμата”, which means “self-acting”

**Definition**

Abstract models for different aspects of computation

- **Sequential computation**
  - Finite memory: Finite state automata
  - Finite memory + Stack: Pushdown automata
  - Unrestricted: Turing machines

- **Concurrent and reactive systems**
  - Nonterminating (ω-words): Büchi automata
  - Nondeterministic (ω-trees): Büchi tree automata

- **Rewriting systems**
  - Terms: Tree automata over ranked trees

- **Semistructured data**
  - XML documents: Tree automata over unranked trees
Finite state automata

Automata for "identifiers" in programming languages
Automata theory

In theoretical computer science, automata theory is

*the study of mathematical properties of abstract computing machines.*

More specifically

- **Expressibility**
  
  *Class of languages* (computational problems) defined in the model
  
  *What the model can and cannot do?*

- **Closure properties**
  
  Closed under the different operations, e.g. union and complement.
  
  *The mathematical structure of the class of languages defined in the model*

- **Decidability and complexity**
  
  Are the decision problems (e.g. nonemptiness, inclusion) decidable?
  
  Can they be solved in PTIME?
  
  *Are there (efficient) algorithms for the statical analysis of the model?*
Outline

1. What is automata theory

2. Why to bother with automata theory?

3. Historical perspective of automata theory

4. About this Course

5. Basic concepts
Why to bother with automata theory?

- Theoretical foundations of various branches of computer science
  - Origin of computer science
    - *Turing machine*
  - Compiler design
    - *Lexical analysis (Finite state automata), Syntactical analysis (Pushdown automata), Code selection (Tree automata)*
  - Foundations of model checking
    - *Büchi automata, Rabin tree automata*
  - Foundations of Web data (XML document) processing
    - *Automata over unranked trees*
  - …

- Abstract and fundamental
  
  Compared to programming languages, automata theory is more abstract, thus ease the mathematical reasoning, but still reflects the essence of computation

- Combinatorial, algorithmic, and challenging
Outline

1. What is automata theory
2. Why to bother with automata theory?
3. Historical perspective of automata theory
4. About this Course
5. Basic concepts
A pioneer of automata theory

Alan Turing (1912-1954)

- Father of computer science
- English logician
- Propose
  Turing machine
  as
  a mathematical model
  of
  computation
- Codebreaker
  for
  German Enigma machine
  in
  World War II
- Many other pioneering work, e.g.
  Turing test
<table>
<thead>
<tr>
<th>Period</th>
<th>Automata Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>1930s</td>
<td>Turing machines (A. Turing)</td>
</tr>
<tr>
<td>1940s -1950s</td>
<td>Finite state automata (W. McCulloch, W. Pitts, S. Kleene, etc.)</td>
</tr>
<tr>
<td></td>
<td>Chomsky hierarchy (N. Chomsky)</td>
</tr>
<tr>
<td></td>
<td>Büchi automata over ω-words (J. R. Büchi)</td>
</tr>
<tr>
<td></td>
<td>Rabin tree automata over ω-trees (M. O. Rabin)</td>
</tr>
<tr>
<td></td>
<td>Tree automata (J. E. Doner, J. W. Thatcher, J. B. Wright, etc.)</td>
</tr>
<tr>
<td>1980s -1990s</td>
<td>ω-automata applied to formal verification</td>
</tr>
<tr>
<td></td>
<td>(M. Vardi, P. Wolper, O. Kupferman, etc.)</td>
</tr>
<tr>
<td>2000s-2010s</td>
<td>Automata over unranked trees applied to XML</td>
</tr>
<tr>
<td></td>
<td>(A. Bruggemann-Klein, M. Murata, D. Wood, F. Neven, etc.)</td>
</tr>
<tr>
<td></td>
<td>Visibly pushdown automata (R. Alur, P. Madhusudan)</td>
</tr>
</tbody>
</table>

Remark: Only include automata models explicitly referred to in this course.
Outline

1. What is automata theory
2. Why to bother with automata theory?
3. Historical perspective of automata theory
4. About this Course
5. Basic concepts
Main goal

An extensive introduction to automata theory, with an emphasis on the developments in the last thirty years.

Organization

Organized according to the different types of structures, Finite words, Infinite words, Finite ranked and unranked trees, Infinite trees

Features

- Extensiveness of the topics
  - In particular, the recent developments are covered
- Emphasis on algorithmic aspects
- Emphasis on the applications

Not included

- Automata models for timed and hybrid systems
  - Timed automata, Hybrid automata
- Automata over infinite alphabets
  - Register automata, data automata, ...
Automata over finite words
- Chomsky hierarchy (A brief recall of the classical automata theory)
  - Turing machines, Linearly-bounded automata,
  - Pushdown automata, Finite state automata
- Finite state automata
  - Nondeterministic versus deterministic, Expressive equivalence with MSO,
  - Myhill-Nerode theorem, Closure properties,
  - Decision problems (Nonemptiness, language Inclusion)
- Visibly pushdown automata
  - Nondeterministic versus deterministic,
  - Closure properties,
  - Decision problems

Automata over infinite words
- Nondeterministic Büchi, Muller, Rabin, Strett, Parity automata and their expressive equivalence
- Expressive equivalence with MSO and WMSO
- Determinization and complementation of Büchi automata
- Decision problems of Büchi automata
• Automata over finite trees
  • Ranked trees
    Bottom-up versus top-down tree automata, Expressive equivalence with MSO, Determinization, Decision problems, Tree-walking automata
  • Unranked trees
    The model, Expressive equivalence with MSO, Determinization, Decision problems

• Automata over infinite trees
  • Büchi tree automata, Rabin tree automata, Parity tree automata, and the comparison of their expressibility
  • Complementation of Rabin tree automata
  • The decision problems
Applications

- Model checking
  - Linear temporal logic and Büchi automata
  - Alternating tree automata, Modal $\mu$-calculus

- XML document processing
  - XPath, DTD, and their relationship with automata over unranked trees
  - Visibly pushdown automata applied to the streaming of XML documents
Grading criteria

- Homework (not so many, do not worry): 20%
- Reading project: 20%
  - Read a paper from a given list and make a presentation
- Reading report: 60%
  - Choose a topic from a given list and make a survey

Remark: The paper list and the topic list will be published later.
References

Course website: http://lcs.ios.ac.cn/~wuzl/teaching.html

Textbooks


Survey articles


Possibly some other papers

Remark: You are not supposed to read all of them.
Outline

1. What is automata theory
2. Why to bother with automata theory?
3. Historical perspective of automata theory
4. About this Course
5. Basic concepts
Finite and infinite words

Fix a finite alphabet $\Sigma$.

- **Finite words**
  
  A sequence of letters from $\Sigma$, i.e. a mapping $w : [n] \rightarrow \Sigma$.
  
  Example: $abab$

- **$\omega$-words**
  
  An $\omega$-sequence of letters from $\Sigma$, i.e. a mapping $w : \mathbb{N} \rightarrow \Sigma$.
  
  Example: $(ab)^\omega$
Finite and infinite trees

- Finite ranked trees
  - Ranked alphabet $\Sigma$: Rank function $\text{rank}() : \Sigma \rightarrow \mathbb{N}$.
  - Tree domain: A nonempty subset $D$ of $\mathbb{N}$ such that
    - if $x_i \in D$, then $x \in D$,
    - if $x_i \in D$, then $x_j \in D$ for any $j \leq i$.
  - Ranked trees: A $\Sigma$-tree is a mapping $t : D \rightarrow \Sigma$ such that
    \[ \forall x \in D, \text{rank}(t(x)) = \max \{ i \mid x_i \in D \} \].
Finite and infinite trees: continued

- Finite unranked trees
  
  **Alphabet $\Sigma$ is unranked**
  
  **Unranked trees: A mapping $t : D \to \Sigma$ (no rank constraints).**

- $\omega$-trees (binary $\omega$-trees)
  
  **A mapping $t : \{0, 1\}^* \to \Sigma$.**

---

**Finite unranked trees**

**Alphabet $\Sigma$ is unranked**

**Unranked trees: A mapping $t : D \to \Sigma$ (no rank constraints).**

- **$\omega$-trees (binary $\omega$-trees)**
  
  **A mapping $t : \{0, 1\}^* \to \Sigma$.**
Formal languages and closure properties

- Formal languages
  
  A set of finite words, finite trees, etc.

- Language-theoretical operations
  
  - Union: $L_1 \cup L_2$,
  
  - Intersection: $L_1 \cap L_2$,
  
  - Complementation: $\Sigma^* \setminus L$, $\Sigma^\omega \setminus L$, ...
  
  - Homomorphism: A mapping $h : \Sigma \rightarrow \Pi \cup \{\varepsilon\}$.

- Closure properties
  
  - Union:
    
    For every pair of automata $A_1$ and $A_2$ in a given model, is $\mathcal{L}(A_1) \cup \mathcal{L}(A_2)$ also accepted by an automaton in the model?
    
    and so on

Example:

*Finite state automata are closed under all Boolean operations (union, intersection and complementation).*
Expressibility and decision problems

- Expressibility: Which languages can be defined in the model?
  Example
  
  \[ \{a^n b^n \mid n \in \mathbb{N}\} \]
  cannot be defined by finite state automata.

- Decision problems
  - Nonemptiness
    
    Given an automaton \(\mathcal{A}\), does \(\mathcal{L}(\mathcal{A}) \neq \emptyset\)?
  
  - Universality
    
    Given an automaton \(\mathcal{A}\), does \(\mathcal{L}(\mathcal{A}) = \Sigma^*\) (for finite words)?
    
    Similarly for other classes of structures
  
  - Language inclusion
    
    Given two automata \(\mathcal{A}_1, \mathcal{A}_2\), does \(\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)\)?