

Two-variable logic over infinite alphabets

Anca Muscholl

LIAFA, Univ. Paris 7 & CNRS

Joint work with:

Mikołaj Bojańczyk (Warsaw, Paris), Claire David (Paris),
Thomas Schwentick (Dortmund) and Luc Segoufin (Paris)

Data

Current algorithmic techniques in program verification and manipulation of XML documents often use data abstraction (word/tree automata based)

- program verification: **abstraction** of variables, parameters, recursion etc. to **finite** range domains or **finite** representations (eg. semi-linear sets)
- XML: documents as ranked/unranked trees with labels from **finite** domain

XML example: Books on the web

```
<doc>
  <fiction>
    <book>
      <title> Der Zauberberg  </title>
      <author> Thomas Mann </author>
      <editor> Fischer </editor>
      <year> ... </year>
      <review> "Nun, wer den Zauberberg kennt
weiss, dass dies kein Buch von fremden
Feenwelten ist, und doch waltet hier
eindeutig Magie."  </review>
    </book>
  </fiction>
```

XML example: Books on the web

```
<CS>
```

```
<book>
```

```
<title> Mathematical logic </title>
```

```
<author> Heinz-Dieter Ebbinghaus </author>
```

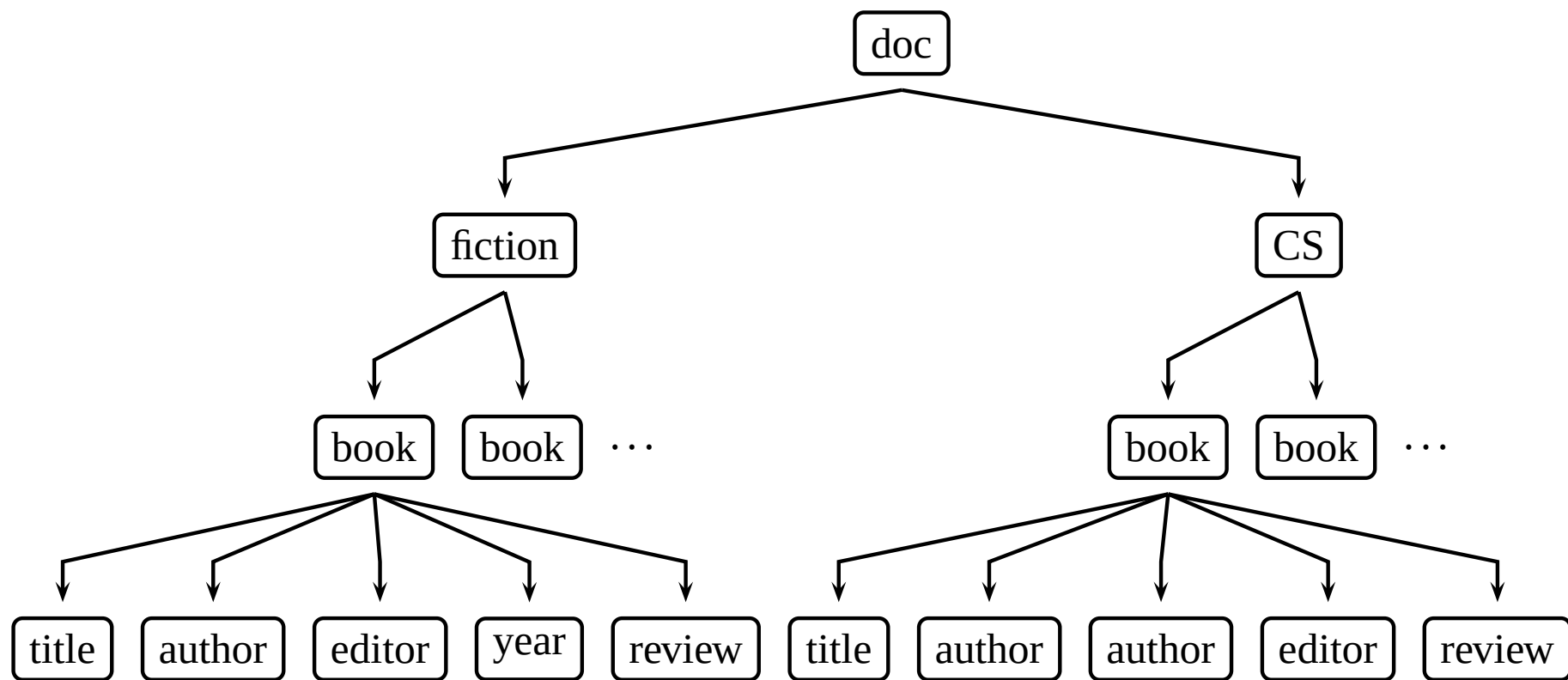
```
<author> Jörg Flum </author>
```

```
<author> Wolfgang Thomas </author>
```

```
<year> 1990 </year>
```

```
<review> "The book remains my text of  
choice for this type of material, and  
I highly recommend it to anyone teaching  
a first logic course at this level  
(J.~of Symb~. Logic)." </review>
```

```
</book> </CS> </doc>
```



XML in essence:

- hierarchically structured through tags
- linear representation of unranked tree
- tree shape defined through DTD: vertical and horizontal (regular) restrictions on tags

XML in essence:

- hierarchically structured through tags
- linear representation of unranked tree
- tree shape defined through DTD: vertical and horizontal (regular) restrictions on tags

... and beyond:

- arbitrary data such as text (leaf nodes), attribute values, references etc.

Queries:

- selection of tree nodes satisfying given property
- query expressed by e.g. XQuery, XPath..., MSO (monadic second-order logic), ...

Queries:

- selection of tree nodes satisfying given property
- query expressed by e.g. XQuery, XPath..., MSO (monadic second-order logic), ...

Consistency:

Given a DTD τ (regular tree language) and a set of constraints on keys/foreign keys $\mathcal{C}$, check whether some XML document $t \in \tau$ exists that satisfies $\mathcal{C}$.

Logic and tree automata

→ FO, MSO talk about (sets of) nodes, tags,
direct/indirect descendants, siblings

Logic and tree automata

- **FO, MSO** talk about (sets of) nodes, tags, direct/indirect descendants, siblings
- **tree automata** evaluate a tree in parallel (bottom-up or top-down) by associating states with nodes

Logic and tree automata

→ **MSO** and bottom-up **tree automata** are equivalent
(Thatcher/Wright '68, Doner '70)

Logic and tree automata

- **MSO** and bottom-up **tree automata** are equivalent (Thatcher/Wright '68, Doner '70)
- No known characterization of (general) **FO** on ranked trees

Logic and tree automata

- **MSO** and bottom-up **tree automata** are equivalent (Thatcher/Wright '68, Doner '70)
- No known characterization of (general) **FO** on ranked trees
- (Benedikt/Segoufin '05) effective characterization of **FO(succ)** on ranked trees

Logic and tree automata

- **MSO** and bottom-up **tree automata** are equivalent (Thatcher/Wright '68, Doner '70)
- No known characterization of (general) **FO** on ranked trees
- (Benedikt/Segoufin '05) effective characterization of **FO(succ)** on ranked trees
- (Cristau/Löding/Thomas FCT'05) effective characterization of **deterministic** top-down trees automata on unranked trees

Questions à la XML

- Determine **decidable** logics that cope with **unrestricted (infinite)** data
- Find **equivalent** tree automata model

Questions à la XML

- Determine **decidable** logics that cope with **unrestricted (infinite)** data
- Find **equivalent** tree automata model

Operations on data: **equality comparisons**

Examples (XPath)

Query: Find books of Thomas with same editor

Unary keys: Attribute A has distinct values

$$\forall x, y. \ x.A = y.A \rightarrow x = y$$

References: Attribute B is a foreign key for attribute A

$$\forall x \exists y. \ x.A = y.B \ \wedge \ \forall x, y. \ \dots$$

Navigation: From node x we can access nodes y_1, y_2 via paths of type p_1, p_2 such that $y_1.A = y_2.B$.

Automata on Data Strings

Data strings:

$(\Sigma \times D)$ -labeled finite strings, with
 Σ finite and D infinite alphabet

Automata with registers (or pebbles) [Kaminski/Francez '94,
Neven/Schwentick/Vianu '04]

- too expressive (emptiness in general undecidable)
- in general incomparable with logics

First-Order Logic with Data

$\text{FO}(\sim, <, +1)$:

- $a(x)$, with $a \in \Sigma$
- order $<$, successor $+1$
- data equality $\sim$

First-Order Logic with Data

FO($\sim$, $<$, $+1$):

- $a(x)$, with $a \in \Sigma$
- order $<$, successor $+1$
- data equality $\sim$

Models: $(\Sigma \times D)$ -labeled strings, (unranked) trees..., with Σ finite and D infinite alphabet

$$x \sim y \quad \text{if} \quad D(x) = D(y)$$

First-Order Logic with Data

FO($\sim, <, +1$):

- $a(x)$, with $a \in \Sigma$
- order $<$, successor $+1$
- data equality $\sim$

Models: $(\Sigma \times D)$ -labeled strings, (unranked) trees..., with Σ finite and D infinite alphabet

$$x \sim y \quad \text{if } D(x) = D(y)$$

In essence: equivalence relation $\sim \rightarrow$ classes (of positions)

FO($\sim, <, +1$) - Examples

Strings with Σ -projection "same number of a 's and b 's"

- Each $\sim$ -class contains precisely one a and one b .

FO($\sim, <, +1$) - Examples

Strings over $\{0, 1\}$ with **same sequence** of 0-values and 1-values:

- Bijection between 0-values and 1-values.
- For each pair of 0-positions $x < y$ and every 1-position z with $x \sim z$ there exists a 1-position $w > z$ with $y \sim w$.

Satisfiability problem

- **Input:** $\text{FO}(\sim, <, +1)$ sentence φ
- **Question:** Is φ satisfiable?

$\text{FO}^k(\sim, <, +1)$: k variables only

Satisfiability problem

- **Input:** $\text{FO}(\sim, <, +1)$ sentence φ
- **Question:** Is φ satisfiable?

$\text{FO}^k(\sim, <, +1)$: k variables only

Satisfiability of $\text{FO}^3(\sim, <)$ formulas on strings is undecidable.

Satisfiability problem

- **Input:** $\text{FO}(\sim, <, +1)$ sentence φ
- **Question:** Is φ satisfiable?

$\text{FO}^k(\sim, <, +1)$: k variables only

Satisfiability of $\text{FO}^3(\sim, <)$ formulas on strings is undecidable.

proof: PCP encoding

Remark A 2-register (2 pebble) automaton can do it too.

Taming the logic: $\text{FO}^2(\sim, <, +1)$

Taming the logic: $\text{FO}^2(\sim, <, +1)$

- Why considering two-variables first-order logic?

Taming the logic: $\text{FO}^2(\sim, <, +1)$

- Why considering two-variables first-order logic?

Core XPath is FO^2 [Gottlob'02, Marx '04]. With one attribute it contains $\text{FO}^2(\sim, <, +1)$.

Taming the logic: $\text{FO}^2(\sim, <, +1)$

- Why considering two-variables first-order logic?

Core XPath is FO^2 [Gottlob'02, Marx '04]. With one attribute it contains $\text{FO}^2(\sim, <, +1)$.

- What properties can be expressed in this logic with data comparisons?

Taming the logic: $\text{FO}^2(\sim, <, +1)$

- Why considering two-variables first-order logic?

Core XPath is FO^2 [Gottlob'02, Marx '04]. With one attribute it contains $\text{FO}^2(\sim, <, +1)$.

- What properties can be expressed in this logic with data comparisons?

Essentially FO^2 + **counting** properties.

$\text{FO}^2(\sim, <, +1)$ properties

Counting properties :

- Same number of a 's and b 's.

$\text{FO}^2(\sim, <, +1)$ properties

Counting properties :

- Same number of a 's and b 's.
- The first a in the **second** class containing an a :

Taming the logic: $\text{FO}^2(\sim, <, +1)$

Counting properties :

- Same number of a 's and b 's.
- The first a in the **second** class containing an a :

$$\dots (a, d_1) \cdots (a, d_1) \cdots (a, d_1) \cdots (a, d_2) \dots$$

- The first a in the **k -th** class containing an a (k fixed).

Two-variable logics

- FO^2 (over graphs) has finite model property (Mortimer'75), **NEXPTIME-complete** (Grädel et al. '97)

Two-variable logics

- FO^2 (over graphs) has finite model property (Mortimer'75), **NEXPTIME-complete** (Grädel et al. '97)
- Over **strings**: $\text{FO}^2(<)$ is equivalent to

Two-variable logics

- FO^2 (over graphs) has finite model property (Mortimer'75), **NEXPTIME-complete** (Grädel et al. '97)
- Over **strings**: $\text{FO}^2(<)$ is equivalent to
 - unary LTL and $\Sigma^2 \cap \Pi^2$ (Etessami, Wilke,...)

Two-variable logics

- FO^2 (over graphs) has finite model property (Mortimer'75), **NEXPTIME-complete** (Grädel et al. '97)
- Over **strings**: $\text{FO}^2(<)$ is equivalent to
 - unary LTL and $\Sigma^2 \cap \Pi^2$ (Etessami, Wilke,...)
 - variety DA (Thérien, Wilke)

Two-variable logics

- FO^2 (over graphs) has finite model property (Mortimer'75), **NEXPTIME-complete** (Grädel et al. '97)
- Over **strings**: $\text{FO}^2(<)$ is equivalent to
 - unary LTL and $\Sigma^2 \cap \Pi^2$ (Etessami, Wilke,...)
 - variety DA (Thérien, Wilke)
 - two way, partially-ordered DFA (Schwentick, Thérien,...)and is **NEXPTIME-complete**.

$\text{FO}^2(\sim, <, +1)$ over strings

Satisfiability of $\text{FO}^2(\sim, <, +1)$ formulas over data strings is decidable. It is equivalent to the reachability problem for Petri nets.

$\text{FO}^2(\sim, <, +1)$ over strings

Satisfiability of $\text{FO}^2(\sim, <, +1)$ formulas over data strings is decidable. It is equivalent to the reachability problem for Petri nets.

User-friendly Petri nets:

multicounter automata = finite automata + **positive counters**

- no test for zero (except at the end)
- accepts with final state + all counters zero
- emptiness is decidable (Mayr, Kosaraju '84)

Satisfiability proof: strings

string projection of data language $L \subseteq (\Sigma \times D)^* =$
projection onto Σ -component

Satisfiability proof: strings

string projection of data language $L \subseteq (\Sigma \times D)^*$ =
projection onto Σ -component

String projections of $\text{FO}^2(\sim, <, +1)$ definable languages
are recognized by multicounter automata.

Idea: $\text{FO}^2(\sim, <, +1)$ can only express that each class
satisfies some regular property and count special classes.

Example: $\{a^n b^n \mid n \geq 0\}$

each class contains precisely one a and one b (to its right) $\Rightarrow$ each **class string** equals ab

Example: $\{a^n b^n \mid n \geq 0\}$

each class contains precisely one a and one b (to its right) $\Rightarrow$ each class string equals ab

Equivalent:

String projection of data language = $\text{Shuffle}(\{ab\}) \cap a^*b^*$

Shuffle of words $w_1, \dots, w_n$:

set of words w that can be colored with n colors s.t. the subsequence of w colored by k is w_k .

Shuffle

example: $aaabbb, aababb \in \text{Shuffle}(\{ab\})$

Shuffle

example: $aaabbb, aababb \in \text{Shuffle}(\{ab\})$

$\text{Shuffle}(L)$ = set of shuffles of words in L

For any regular language L , the language $\text{Shuffle}(L)$ is recognized by a multicounter automaton (Gischer '81).

Shuffle

example: $aaabbb, aababb \in \text{Shuffle}(\{ab\})$

$\text{Shuffle}(L)$ = set of shuffles of words in L

For any regular language L , the language $\text{Shuffle}(L)$ is recognized by a multcounter automaton (Gischer '81).

Conversely: the set of accepting runs of a multcounter automaton can be expressed as $\text{Shuffle}(L) \cap R$.

$\text{FO}^2(\sim, +1)$ on strings

String projections of $\text{FO}^2(\sim, +1)$ definable languages are recognized by linear constraint automata (LCA).

LCA: finite automata with acceptance defined by linear constraints on the number of transitions

$\text{FO}^2(\sim, +1)$ on strings

String projections of $\text{FO}^2(\sim, +1)$ definable languages are recognized by linear constraint automata (LCA).

LCA: finite automata with acceptance defined by linear constraints on the number of transitions

→ formula yields a 2exp-sized LCA

Emptiness of linear constraint automata is NP-complete.
[Seidl/Schwentick/M./Habermehl '04]

Complexity: String case

- polynomial-time reduction from emptiness of multicounter to satisfiability
- Emptiness of multicounter automata (reachability of Petri nets): lower bound EXPSpace, no elementary upper bound known
- satisfiability of $\text{FO}^2(\sim, <)$ is NEXPTIME-complete
- satisfiability of $\text{FO}^2(\sim, +1)$ is in 2NEXPTIME

Data strings: related work

- Data automata of Bouyer/Petit/Thérien '03: 1-way automata, strictly weaker
- Closest to our setting: temporal logic with [freeze operator](#) [Demri/Lazic/Nowak Time'05]

$\text{FO}^2(\sim, <, +1)$ on unranked trees

- Reduction from $\text{FO}^2(\sim, <, +1)$ on trees to emptiness of **tree multcounter automata**: **open problem** [de Groote et al., LICS'04]
- Reduction from emptiness of **tree multcounter automata** to satisfiability of $\text{FO}^2(\sim, <, +1)$.

$\text{FO}^2(\sim, +1)$ on unranked trees

$\text{FO}^2(\sim, +1)$ on unranked (finite) trees is decidable

$\text{FO}^2(\sim, +1)$ on unranked trees

$\text{FO}^2(\sim, +1)$ on unranked (finite) trees is decidable

Idea:

- $\text{FO}^2(\sim, +1)$ has small models (= few large connected zones with same data value)
- Small models are recognized by linear constraint tree automata (= bottom-up unranked TA with linear constraints over transitions frequencies).
- Emptiness of linear constraint tree automata is decidable.

Unranked trees: related work

- (Benedikt, Fan, Geerts PODS'05) Satisfiability of XPath with data equality tests is decidable, but **no** horizontal navigation (and either no negation or no vertical order).
- (Arenas, Fan, Libkin '05) **Consistency** problem for unary/foreign keys with a DTD is decidable: included into $\text{FO}^2(\sim, +1)$
- (Kieroński, Otto LICS'05) Satisfiability for arbitrary structures with **3 equivalence relations** is undecidable (but: XPath cannot even do joins, such as $x.A = y.A \wedge x.B = y.B$).

Open problems and conclusion

- 1-register/pebble automata equivalent to $\text{FO}^2(\sim, <, +1)$?
- extension to $\text{FO}^2(\sim, <, +k)$?
- link with temporal logics with data [Demri/Lazic '05]?
- applications to the verification of parametrized systems?

Open problems and conclusion

- 1-register/pebble automata equivalent to $\text{FO}^2(\sim, <, +1)$?
- extension to $\text{FO}^2(\sim, <, +k)$?
- link with temporal logics with data [Demri/Lazic '05]?
- applications to the verification of parametrized systems?

Thank you for your attention!