Is the generic algorithm for first-order model-checking automatic structures optimal ?

Antoine Durand-Gasselin Peter Habermehl

LIAFA, Université Paris Diderot

September 21^{st}, 2013

Contents

First-Order Logics

Interpreted relational structure

the domain

a set of symbols with arity

$$\mathcal{A} = (\, D, \mathcal{S}, (P^{\mathcal{A}})_{P \in \mathcal{S}} \,)$$

for $P_i \in \mathcal{S}$, $P_i^{\mathcal{A}} \subseteq D^{ar_{P_i}}$

interpretation of predicates over D

First-Order Logics

Interpreted relational structure

the domain

a set of symbols with arity

$$\mathcal{A} = (D, \mathcal{S}, (P^{\mathcal{A}})_{P \in \mathcal{S}})$$

for $P_i \in \mathcal{S}$, $P_i^{\mathcal{A}} \subseteq D^{ar_{P_i}}$

interpretation of predicates over D

First-order formulas

Defined inductively: $\varphi ::= (x_i = x_j) \mid P(x_1, \ldots, x_{ar_P}) \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \exists x \cdot \varphi$

First-Order Logics

Interpreted relational structure

the domain — a set of symbols with arity

$$\mathcal{A} = (D, \mathcal{S}, (P^{\mathcal{A}})_{P \in \mathcal{S}})$$

interpretation of predicates over D

for $P_i \in \mathcal{S}$, $P_i^{\mathcal{A}} \subseteq D^{ar_{P_i}}$

First-order formulas

Defined inductively: $\varphi ::= (x_i = x_j) \mid P(x_1, \ldots, x_{ar_P}) \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \exists x \cdot \varphi$

First-order model-checking problem over $\mathcal{A}$

Input: First-order formula over some signature $\mathcal{S}$

Output: Whether the formula is satisfiable in the structure

Some structures can be presented with automata

- The domain D is a regular language over alphabet Σ (accepted by A_D)
- Predicates are also regular languages... synchronously regular relations

Some structures can be presented with automata

- The domain D is a regular language over alphabet Σ (accepted by A_D)
- Predicates are also regular languages... synchronously regular relations
  - How can an automaton test whether $P(w_1, w_2, w_3)$ holds ?

Some structures can be presented with automata

- The domain D is a regular language over alphabet Σ (accepted by A_D)
- Predicates are also regular languages... synchronously regular relations
  - How can an automaton test whether $P(w_1, w_2, w_3)$ holds ?
  - ... with an automaton over alphabet Σ^3

Some structures can be presented with automata

- The domain D is a regular language over alphabet Σ (accepted by A_D)
- Predicates are also regular languages... synchronously regular relations
  - How can an automaton test whether $P(w_1, w_2, w_3)$ holds ?
  - ... with an automaton over alphabet $(\Sigma \cup \{\diamond\})^3$
  - we need an additional padding symbol if words don't have same length: $aa \otimes \varepsilon \otimes abbaab = (\diamond, \diamond, a)(\diamond, \diamond, b)(\diamond, \diamond, b)(\diamond, \diamond, a)(a, \diamond, a)(a, \diamond, b)$

Some structures can be presented with automata

- The domain D is a regular language over alphabet Σ (accepted by A_D)
- Predicates are also regular languages... synchronously regular relations
  - How can an automaton test whether $P(w_1, w_2, w_3)$ holds ?
  - ... with an automaton over alphabet $(\Sigma \cup \{\diamond\})^3$
  - we need an additional padding symbol if words don't have same length: $aa \otimes \varepsilon \otimes abbaab = (\diamond,\diamond,a)(\diamond,\diamond,b)(\diamond,\diamond,b)(\diamond,\diamond,a)(a,\diamond,a)(a,\diamond,b)$

Theorem [Hodgson, 1982, Khoussainov and Nerode, 1994]

For automatic structures, any FO relation is synchronously regular

For each formula φ, there exists an automaton (denoted A_φ) that accepts exactly the set of solutions of φ.

Some structures can be presented with automata

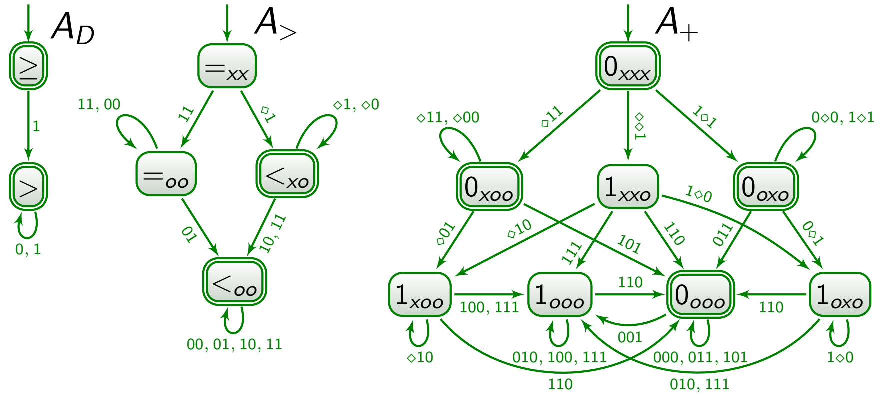
- The domain D is a regular language over alphabet Σ (accepted by A_D)
- Predicates are also regular languages... synchronously regular relations
  - How can an automaton test whether $P(w_1, w_2, w_3)$ holds ?
  - ... with an automaton over alphabet $(\Sigma \cup \{\diamond\})^3$
  - we need an additional padding symbol if words don't have same length: $aa \otimes \varepsilon \otimes abbaab = (\diamond, \diamond, a)(\diamond, \diamond, b)(\diamond, \diamond, b)(\diamond, \diamond, a)(a, \diamond, a)(a, \diamond, b)$

Theorem [Hodgson, 1982, Khoussainov and Nerode, 1994]

For automatic structures, any FO relation is synchronously regular

For each formula φ, there exists an automaton (denoted A_φ) that accepts exactly the set of solutions of φ.

Example: Automatic Presentation of Presburger Arithmetic



Example: $(18, 170, 188) \in L(A_+)$
encoded by $(\diamond 11, \diamond 00, \diamond 11, 101, 011, 001, 110, 000)$

◇	◇	◇	1	0	0	1	0
1	0	1	0	1	0	1	0
1	0	1	1	1	1	0	0

Contents

Construction of the automaton A_φ

We use the correspondance between logical connectives and automata operations:

Construction of the automaton A_φ

We use the correspondance between logical connectives and automata operations:

- We have by definition automata for atomic formulas

Construction of the automaton A_φ

We use the correspondance between logical connectives and automata operations:

- We have by definition automata for atomic formulas
- Conjunction corresponds to language intersection:
  Solutions of $\varphi \wedge \psi$ are $L(A_\varphi) \cap L(A_\psi)$

Construction of the automaton A_φ

We use the correspondance between logical connectives and automata operations:

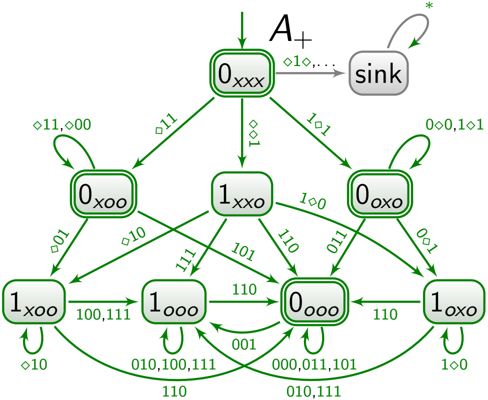
- We have by definition automata for atomic formulas
- Conjunction corresponds to language intersection:
  Solutions of $\varphi \wedge \psi$ are $L(A_\varphi) \cap L(A_\psi)$
- Negation corresponds to language complementation:
  Solutions of $\neg\varphi$ are tuples of elements of the domain among $\overline{L(A_\varphi)}$

Construction of the automaton A_φ

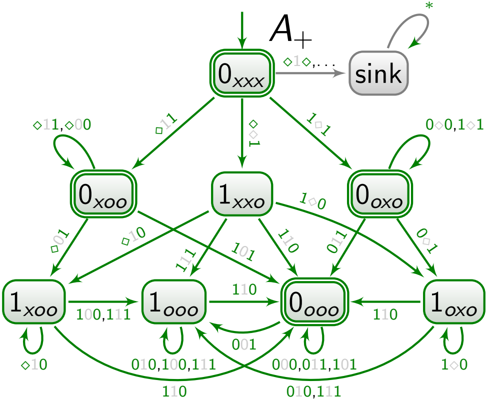
We use the correspondance between logical connectives and automata operations:

- We have by definition automata for atomic formulas
- Conjunction corresponds to language intersection:
  Solutions of $\varphi \wedge \psi$ are $L(A_\varphi) \cap L(A_\psi)$
- Negation corresponds to language complementation:
  Solutions of $\neg\varphi$ are tuples of elements of the domain among $\overline{L(A_\varphi)}$
- Existential quantification corresponds to language projection:
  From a language over $(\Sigma \cup \{\diamond\})^r$ we need to accept accept a language over $(\Sigma \cup \{\diamond\})^{r-1}$, erasing the track corresponding to the quantified variable.

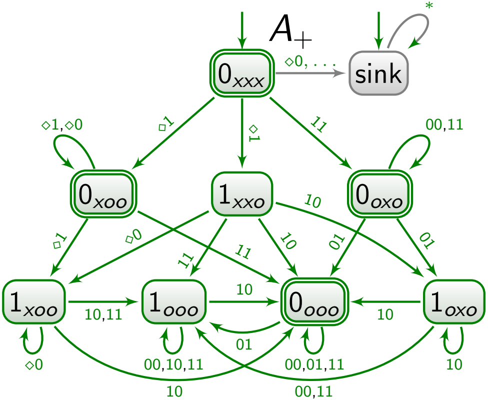
Building $A_{\exists y.x+y=z}$



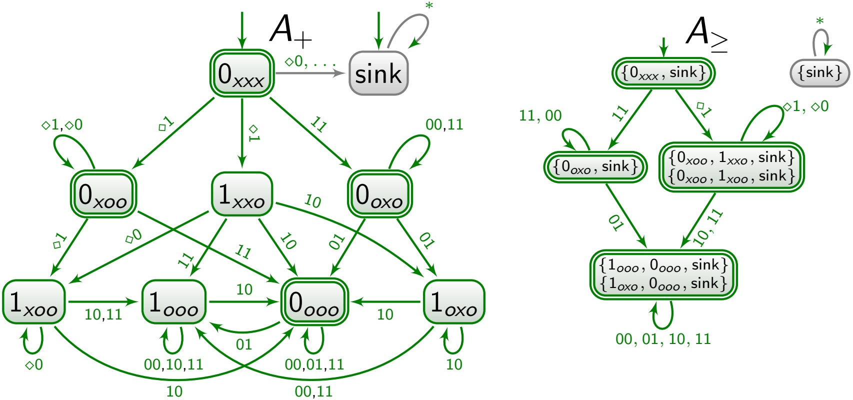
Building $A_{\exists y.x+y=z}$



Building $A_{\exists y.x+y=z}$



Building $A_{\exists y.x+y=z}$



Is this generic algorithm optimal ?

Model-Checking through inductive automaton construction

- Model-checking φ is reduced to checking emptiness of A_φ
- What is the complexity of this automaton construction?

We will characterize the *deterministic time complexity* of this construction.

Is this generic algorithm optimal ?

Model-Checking through inductive automaton construction

- Model-checking φ is reduced to checking emptiness of A_φ
- What is the complexity of this automaton construction?

We will characterize the *deterministic time complexity* of this construction.

A naive analysis

Theorem

- Some automatic structures have non-elementary first-order theory
- The complexity of the inductive automaton construction is non-elementary

Indeed a quantifier alternation corresponds to a language projection and a complementation: it may lead to an exponential blow-up.

Is this generic algorithm optimal ?

Model-Checking through inductive automaton construction

- Model-checking φ is reduced to checking emptiness of A_φ
- What is the complexity of this automaton construction?

We will characterize the *deterministic time complexity* of this construction.

A naive analysis

Theorem

- Some automatic structures have non-elementary first-order theory
- The complexity of the inductive automaton construction is non-elementary

Indeed a quantifier alternation corresponds to a language projection and a complementation: it may lead to an exponential blow-up.

Yes

Is this generic algorithm optimal ?

Model-Checking through inductive automaton construction

- Model-checking φ is reduced to checking emptiness of A_φ
- What is the complexity of this automaton construction?

We will characterize the *deterministic time complexity* of this construction.

A naive analysis

Theorem

- Some automatic structures have non-elementary first-order theory
- The complexity of the inductive automaton construction is non-elementary

Indeed a quantifier alternation corresponds to a language projection and a complementation: it may lead to an exponential blow-up.

Yes, but what if we fix the structure ?

Contents

Saturated structure

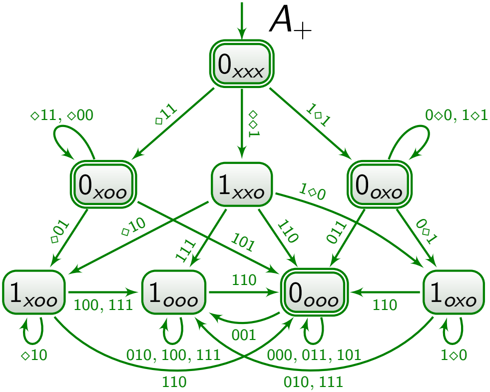
Given an automatic presentation, we define its saturated structure:

- Its domain is Σ^*
- For each state in each automaton we define a predicate (with the corresponding arity) that states reachability of that state

Saturated structure

Given an automatic presentation, we define its saturated structure:

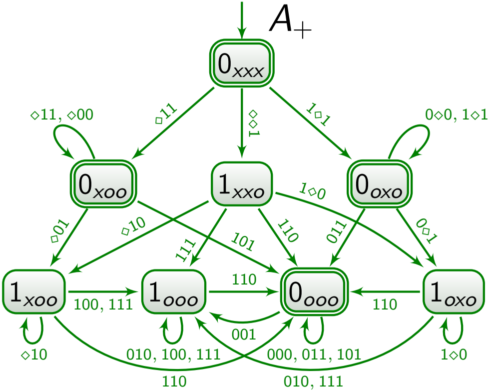
- Its domain is Σ^*
- For each state in each automaton we define a predicate (with the corresponding arity) that states reachability of that state



Saturated structure

Given an automatic presentation, we define its saturated structure:

- Its domain is Σ^*
- For each state in each automaton we define a predicate (with the corresponding arity) that states reachability of that state



$0_{xxx} = \{(\varepsilon, \varepsilon, \varepsilon)\}$
$0_{xoo} = \{(\varepsilon, w, w) | w \in 1\Sigma^*\}$
$1_{xoo} = \{(0, x, x + 1)\}$
$0_{ooo} = \{(x, y, x + y)\}$
$1_{ooo} = \{(x, y, x + y + 1)\}$

Remark that words of the form $0\Sigma^*$ are elements in the domain of the saturated structure but are not in any relation.

The saturated structure exhibits the complexity

Theorem: [Durand-Gasselin and Habermehl, 2012]

The deterministic time complexity of this automaton construction is the same as for model-checking the saturated structure

Remark

The first-order theory of the saturated structure of that Presburger Arithmetic presentation is no harder than Presburger Arithmetic

Corrolary

The inductive construction of an automaton accepting solutions of a Presburger formula is in 3EXPTIME

This algorithm is optimal in this case □

Bonus: the automaton has at most a triple exponential number of states

Conclusion

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Open questions

- Is the saturated structure harder than the presented structure ?

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Open questions

- Is the saturated structure harder than the presented structure ?
  - Remark: there has always been a “good” presentation

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Open questions

- Is the saturated structure harder than the presented structure ?
  - Remark: there has always been a “good” presentation
- Is it possible to establish space-constrained algorithm ?

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Open questions

- Is the saturated structure harder than the presented structure ?
  - Remark: there has always been a “good” presentation
- Is it possible to establish space-constrained algorithm ?
  - Informal remark: we have polynomial time language projection, we would need “NL language projection”

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Open questions

- Is the saturated structure harder than the presented structure ?
  - Remark: there has always been a “good” presentation
- Is it possible to establish space-constrained algorithm ?
  - Informal remark: we have polynomial time language projection, we would need “NL language projection”
- Can we decide the complexity of an automatic presentation ?

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Open questions

- Is the saturated structure harder than the presented structure ?
  - Remark: there has always been a “good” presentation
- Is it possible to establish space-constrained algorithm ?
  - Informal remark: we have polynomial time language projection, we would need “NL language projection”
- Can we decide the complexity of an automatic presentation ?
  - Are there automatic structures with elementary first-order theory harder than 3EXPTIME ?

Conclusion

An effective criterion

The complexity of the saturated structure characterizes the complexity of the automata inductive construction.

Open questions

- Is the saturated structure harder than the presented structure ?
  - Remark: there has always been a "good" presentation
- Is it possible to establish space-constrained algorithm ?
  - Informal remark: we have polynomial time language projection, we would need "NL language projection"
- Can we decide the complexity of an automatic presentation ?
  - Are there automatic structures with elementary first-order theory harder than 3EXPTIME ?

Thank you for your attention !

References

Durand-Gasselin, A. and Habermehl, P. (2012).
Ehrenfeucht-fraïssé goes elementarily automatic for structures of bounded degree.
In *STACS*, volume 14 of *LIPIcs*, pages 242–253. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Hodgson, B. R. (1982).
On direct products of automaton decidable theories.
Theoretical Computer Science, 19(3):331–335.

Khoussainov, B. and Nerode, A. (1994).
Automatic presentations of structures.
In *Logical and Computational Complexity*, volume 960 of *LNCS*, pages 367–392. Springer.