Deciding S1S

Down the Rabbit Hole and Through the Looking Glass

Vojta Havlena **Ondra Lengál** Bára Šmahlíková

Brno University of Technology
Czech Republic

NETYS'21

S1S

S1S:

- second-order monadic logic of one successor:
  - a logic over the structure $(\mathbb{N}, S)$
    - S is a unary function denoting successor, e.g. $S(S(0)) = 2$
  - quantification over individual and set variables
- one of the first logics with automata-based decision procedure [Büchi'62]
  - equivalent to Büchi automata (i.e., ω-regular languages)
- **NONELEMENTARY** complexity lower bound

S1S

S1S:

- second-order monadic logic of one successor:
  - a logic over the structure $(\mathbb{N}, S)$
    - S is a unary function denoting successor, e.g. $S(S(0)) = 2$
  - quantification over individual and set variables
- one of the first logics with automata-based decision procedure [Büchi'62]
  - equivalent to Büchi automata (i.e., ω-regular languages)
- **NONELEMENTARY** complexity lower bound

Uses:

- system specification & verification
  - more expressive and concise than LTL
  - model checking $\mathcal{M} \models \varphi$
- reasoning about natural numbers
- general logic for encoding other logics
  - WS1S, Presburger arithmetic, first-order theory of ω-automatic structures, . . .
  - first-order theory of Sturmian words over Presburger arithmetic

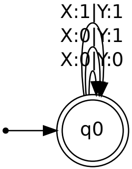
This paper

This paper:

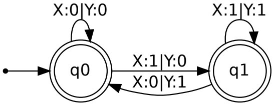
- implementation of the classical decision procedure of Büchi:
  - translation of φ to a Büchi automaton $\mathcal{A}_\varphi$
  - satisfiability — testing language emptiness of $\mathcal{A}_\varphi$
- evaluating efficiency of various algorithms for handling Büchi automata
- comparison with the loop-DFA (L-DFA) based decision procedure for S1S
  - S. Barth. *Deciding Monadic Second Order Logic over ω-Words by Specialized Finite Automata*. IFM'16. Springer.
    - based on H. Calbrix, M. Nivat, and A. Podelski. *Ultimately periodic words of rational ω-languages*. MFPS'93. Springer.

Deciding S1S

- atomic predicates:

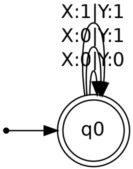


(a) $X \subseteq Y$

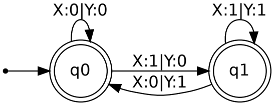


(b) $Y = Succ(X)$

- atomic predicates:



(a) $X \subseteq Y$



(b) $Y = Succ(X)$

- composed predicates:

$\neg\varphi$	Complement $\mathcal{A}_\varphi$
$\varphi_1 \wedge \varphi_2$	Intersect $\mathcal{A}_{\varphi_1}$ and $\mathcal{A}_{\varphi_2}$
$\varphi_1 \vee \varphi_2$	Union of $\mathcal{A}_{\varphi_1}$ and $\mathcal{A}_{\varphi_2}$
$\exists X.\varphi$	Projection of X from $\mathcal{A}_\varphi$

Implementation

- tool Alice in Python
- simple LISP-like input format

$$0 \in X \quad \wedge \quad \forall Y \forall Z (Z = Succ(Y))$$

$$\rightsquigarrow$$

```
(and (zeroin X) (forall Y (forall Z (succ Z Y))))
```

- intersection, union, projection, emptiness checking $\mapsto$ standard algorithms
- complementation $\mapsto$
  - Schewe's rank-based algorithms [Schewe'09]
  - determinization based algorithm in Spot
- simulation-based reductions

Experiments

no.	Formula	State count		
		BA - Schewe	BA - Spot	L-DFA[1]
1	$(x \in Y \land x \notin Z) \lor (x \in Z \land x \notin Y)$	**2**	**2**	9
3	$after(X, Y) := \forall x.(x \in X \Rightarrow \exists y.(y > x \land y \in Y))$	5	**3**	9
4	$fair(X, Y) := after(X, Y) \land after(Y, X)$	24	**5**	9
5	$\forall X.(fair(X, Y) \Rightarrow fair(Y, Z))$	OOM	21	**14**
6	$suc(x, y) := x < y \land \forall z.(\neg x < z \lor \neg z < y))$	**3**	**3**	10
18	$offset(X, Y) := \forall i \forall j.(suc(i, j) \land i \in X \Rightarrow j \in Y)$	**2**	**2**	11
19	$offset(X, Y) \land offset(Y, Z) \land offset(Z, X)$	**8**	**8**	107
20	$offset(V, W) \land offset(W, X) \land offset(X, Y) \land offset(Y, Z) \land offset(Z, V)$	**32**	**32**	2331
21	$\exists Y.(offset(X, Y) \land offset(Y, Z))$	**4**	**4**	29
22	$insm(i, j, U, V, W) := (j \in U \Rightarrow i \in V \lor i \in W)$	**8**	**8**	15
23	$\forall i \forall j(suc(i, j) \Rightarrow insm(i, j, U, V, Z) \land insm(i, j, V, X, Y) \land insm(i, j, X, Y, V) \land insm(i, j, Y, Z, X) \land insm(i, j, Z, U, Y))$	OOM	TO	**198**
24	$\forall x \forall y.(x < y \land y \in X \land y \in Y)$	**3**	**3**	9
26	$\forall x \forall y.(x < y \land y \in X \land y \in Y) \land \forall x \forall y.(x < y \land y \in X \land y \notin Y) \land \forall x \forall y.(x < y \land y \notin X \land y \in Y) \land \forall x \forall y.(x < y \land y \notin X \land y \notin Y)$	21	**11**	18

- SPOT's complementation usually better than basic Schewe
- ALICE: usually less states (but handling Büchi automata is harder)
- #19 & #20: much better scalability

[1] S. Barth. *Deciding Monadic Second Order Logic over ω-Words by Specialized Finite Automata*. IFM'16.