Synthesis of Data Words Transducers

Léo Exibard[1,2] Pierre-Alain Reynier[1] Emmanuel Filiot[2]

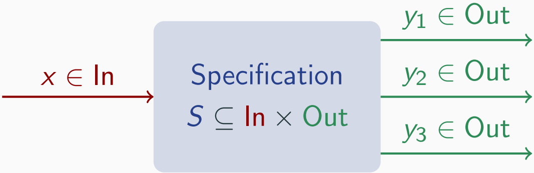
Wednesday, September 11th, 2019

[1]Laboratoire d'Informatique et des Systèmes
Aix-Marseille Université
France

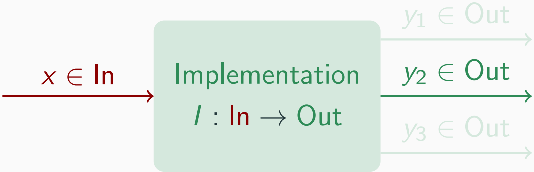
[2]Méthodes Formelles et Vérification
Université libre de Bruxelles
Belgium

Statement of the Problem

Given

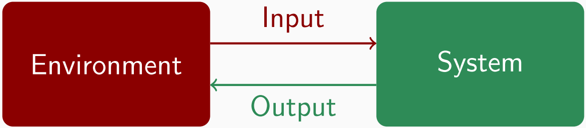


Does there exist



Reactive Synthesis

Reactive systems



Interaction $\rightsquigarrow$ $i_1\, o_1\, i_2\, o_2\, i_3\, o_3 \ldots$

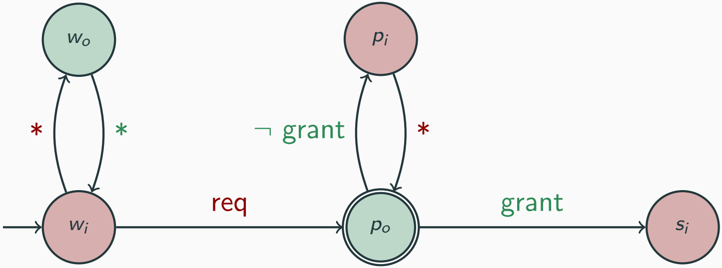
$$\begin{aligned} \mathsf{In} &= I^\omega \\ \mathsf{Out} &= O^\omega \end{aligned}$$

Goal

Generate a system from a specification

$? \parallel \mathsf{Env} \models \mathsf{Specification}$

Finite Automata



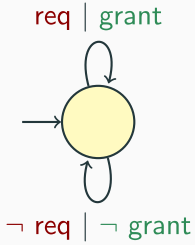
A Universal co-Büchi Automaton checking that every request is eventually granted.

Relation recognised by A

$$\mathcal{R}(A) = \{(i_1 i_2 \ldots, o_1 o_2 \ldots) \mid i_1 o_1 i_2 o_2 \cdots \in \mathcal{L}(A)\}$$

The Classical Setting: Implementations

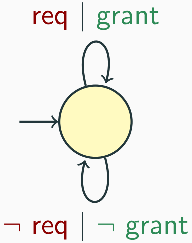
Finite Transducers



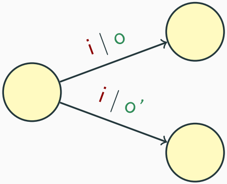
- Automata with outputs
- Deterministically outputs a letter on reading a letter
- No accepting states

The Classical Setting: Implementations

Finite Transducers



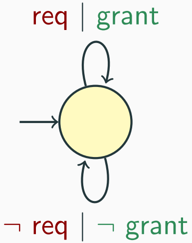
Sequentiality



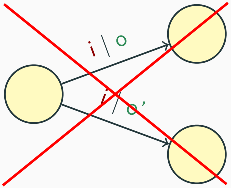
- Automata with outputs
- Deterministically outputs a letter on reading a letter
- No accepting states

The Classical Setting: Implementations

Finite Transducers



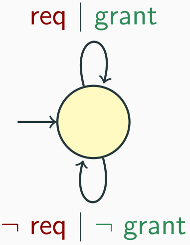
Sequentiality



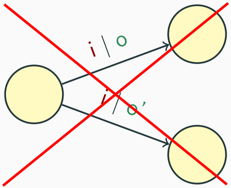
- Automata with outputs
- Deterministically outputs a letter on reading a letter
- No accepting states

The Classical Setting: Implementations

Finite Transducers



Sequentiality



- Automata with outputs
- Deterministically outputs a letter on reading a letter
- No accepting states

Theorem

The synthesis of Sequential Transducers from Nondeterministic Finite Automata is ExpTime-c [Büchi and Landweber, 1969].

From finite alphabets to data words

Motivating example

Every request of client i is eventually granted:

$$\bigwedge_{i \in C} G\Big(req(i) \rightarrow F\big(grant(i)\big)\Big)$$

Limitation

Input and output alphabets are assumed to be small (*finite*) sets.

- Classical setting: C finite
- Our setting: C infinite

How to Represent Executions? Data Words

- Sequences of pairs $(a, d) \in \Sigma \times \mathcal{D}$
- Σ finite alphabet of *labels*
- $\mathcal{D}$ infinite set of *data*

1	4	2	2	3	1	5	3
req	¬grt	req	grt	req	grt	¬req	grt

- $\Sigma = \{\mathsf{req}, \mathsf{grt}, \neg\mathsf{req}, \neg\mathsf{grt}\}$
- $\mathcal{D} = \mathbb{N}$

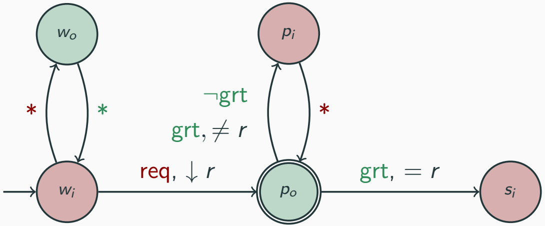
Extending Automata to Data Words: Register Automata

Finite automata with a finite set **R** of registers

- Store data
- Test register content

Transitions $q \xrightarrow{\sigma,\varphi,A} q'$

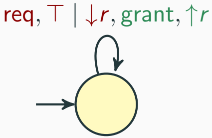
- σ label
- $\varphi \subseteq R$ tests
- A registers assigned d



An URA checking that every request is eventually granted.

Sequential Register Transducers

- Transitions $q \xrightarrow{i,\varphi \ \mid \ A,\ o,\ r_{\text{out}}} q'$
    - i input letter, o output letter
    - φ test over d_{in}
    - A registers assigned d_{in}
    - r_{out} register whose content is output
- Sequentiality: tests are mutually exclusive



A register transducer immediately granting each request.

Synthesis of Register Transducers

Unbounded Synthesis Problem

Input: S a register automaton

Output:
- M a register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Synthesis of Register Transducers

Unbounded Synthesis Problem

Input: S a register automaton

Output:
- M a register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Theorem

The unbounded synthesis problem is undecidable for S given as a Nondeterministic Register Automaton. 🙁

Synthesis of Register Transducers

Unbounded Synthesis Problem

Input: S a register automaton

Output:
- M a register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Theorem

The unbounded synthesis problem is undecidable for S given as a Nondeterministic Register Automaton.

➔ Universality of NRA over finite words is undecidable

Synthesis of Register Transducers

Unbounded Synthesis Problem

Input: S a register automaton

Output:
- M a register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Theorem

The unbounded synthesis problem is undecidable for S given as a Universal Register Automaton. 

- ➔ Slightly more complex proof
- ➔ Open question in [Khalimov et al., 2018]

Synthesis of Register Transducers

Unbounded Synthesis Problem

Input: S a register automaton

Output:
- M a register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Theorem

The unbounded synthesis problem is decidable for S given as a Deterministic Register Automaton.

- ➔ Reduce to bounded synthesis
- ➔ S is realisable by a register transducer iff it is realisable by a $|R_S|$-registers transducer

Bounded Synthesis of Register Transducers

Bounded Synthesis Problem

Input: S a register automaton, k a number of registers

Output:
- M a k-register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Results

- Still undecidable for S nondeterministic (even for $k = 1$) 

Bounded Synthesis Problem

Input: S a register automaton, k a number of registers

Output:
- M a k-register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Results

- Still undecidable for S nondeterministic (even for $k = 1$)
- Decidable for S universal [Khalimov et al., 2018, we provide an alternative, simpler proof]

Bounded Synthesis of Register Transducers

Bounded Synthesis Problem

Input: S a register automaton, k a number of registers

Output:
- M a k-register transducer s.t. $M \models S$ if it exists
- **No** otherwise

Results

- Still undecidable for S nondeterministic (even for $k = 1$)
- Decidable for S universal [Khalimov et al., 2018, we provide an alternative, simpler proof]
- Decidable for S nondeterministic test-free
  - ➔ Test-free: cannot test equality between input data

Conclusion

Main results

Synthesis	DRA	NRA	URA	$\mathrm{NRA}_{\mathrm{tf}}$
Bounded	ExpTime	Undecidable	2ExpTime	2ExpTime
Unbounded			Undecidable	Open

Ongoing work

- Complexity lower bounds
- For S functions, decision of sequentiality and continuity
- Decision of functionality

Future work

- Synthesis from logical specifications

Proof Idea: Reduce to a Finite Alphabet

Abstract actions

- Input actions: $(i, tst) \in \Sigma_{\text{in}} \times 2^k$
- Output actions: $(asgn, o, r_{\text{out}}) \in 2^k \times \Sigma_{\text{out}} \times 2^k$

➔ $w \in (\Sigma \times \mathcal{D})^\omega$ is compatible with $\mathbf{a} = a_1 a_2 \ldots$ iff $a_1 a_2 \ldots$ can be performed on reading w.

Example

Sequence	$(a, \varnothing)$	$(\{r_1\}, b, \{r_1\})$	$(a, \varnothing)$	$(\{r_2\}, b, \{r_1\})$	$(a, \{r_1\})$
Word	(a,1)	(b,1)	(a,2)	(b,1)	(a,1)
Registers	(0,0)	(1,0)	(1,0)	(1,2)	(1,2)

Proof Idea: Reduce to a Finite Alphabet

Proposition

- S is realisable by a k-register transducer iff
- $W_{S,k} = \{\mathbf{a} \text{ abstract sequence} \mid \mathrm{Comp}(\mathbf{a}) \subseteq S\}$ is realisable by a (register-free) finite transducer

Proposition

- $W_{S,k}$ is ω-regular for S Universal Register Automaton
- $W_{S,k}$ is ω-regular for S Nondeterministic test-free Register Automaton

Bibliography i

Büchi, J. R. and Landweber, L. H. (1969).
Solving Sequential Conditions by Finite-State Strategies.
Transactions of the American Mathematical Society, 138:295–311.

Khalimov, A., Maderbacher, B., and Bloem, R. (2018).
Bounded Synthesis of Register Transducers.
In *Automated Technology for Verification and Analysis, 16th International Symposium, ATVA 2018, Los Angeles, October 7-10, 2018. Proceedings*.