Recent advances in register-bounded synthesis

Léo Exibard, Emmanuel Filiot, *Ayrat Khalimov*

MVF seminar, May 2022

Outline

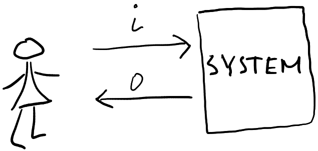
What is reg-bounded synthesis?

- motivation
- register automata
- the synthesis problem
- history
- known and recent advances

Recent advances:

- reg-bounded synthesis:
  from $(\mathbb{N}, <)$ to *regapprox* domains
- reducibility between domains

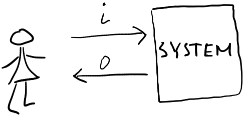
Synthesis



$$i_0 o_0 i_1 o_1 \ldots \in (I \cdot O)^\omega$$

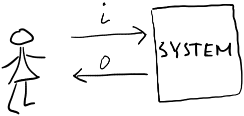
Synthesis problem:
$\rightarrow$ specification language $\subseteq (I \cdot O)^\omega$
$\leftarrow$ transducer whose every interaction $\in$ spec, else UNREALIZABLE

Why Consider Data Transducers?



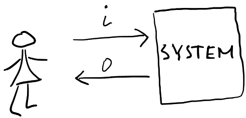
data buffer:
"always relay input data to the output"

Why Consider Data Transducers?



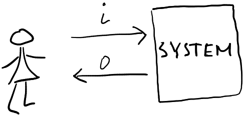
priority arbiter:
"if a process requests an access, it is eventually granted to this or higher-ID process"

SYNTHESIS OF DATA TRANSDUCERS



Synthesis problem:
given specification $S \subseteq (I \cdot O)^\omega$, return transducer with $L(T) \subseteq S$ or UNREAL

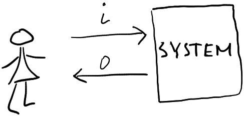
Synthesis of Data Transducers



Synthesis problem:
given specification $S \subseteq (I \cdot O)^\omega$, return transducer with $L(T) \subseteq S$ or UNREAL

Finite-alphabet case:
spec: LTL, ω-reg expressions, MSO, nondet/univ/det automata

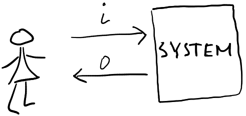
Synthesis of Data Transducers



Synthesis problem:
given specification $S \subseteq (I \cdot O)^\omega$, return transducer with $L(T) \subseteq S$ or UNREAL

Finite-alphabet case:
spec: LTL, $\underbrace{\omega\text{-reg expressions, MSO, nondet/univ/det automata}}_{\text{equally expressive}}$

Synthesis of Data Transducers



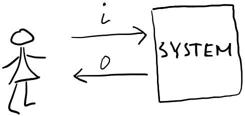
Synthesis problem:
given specification $S \subseteq (I \cdot O)^\omega$, return transducer with $L(T) \subseteq S$ or UNREAL

Finite-alphabet case:
spec: LTL, $\underbrace{\text{$\omega$-reg expressions, MSO, nondet/univ/det automata}}_{\text{equally expressive}}$

$<$

Data case:
spec: FO with $>_d$, Constraint LTL, LTL with freeze quantifier, pebble automata, variable automata, register automata ...

Synthesis of Data Transducers



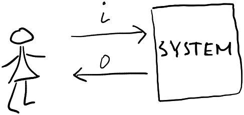
Synthesis problem:
given specification $S \subseteq (I \cdot O)^\omega$, return transducer with $L(T) \subseteq S$ or UNREAL

Finite-alphabet case:
spec: LTL, $<$ $\underbrace{\omega\text{-reg expressions, MSO, nondet/univ/det automata}}_{\text{equally expressive}}$

Data case:
spec: FO with $>_d$, Constraint LTL, LTL with freeze quantifier, pebble automata, variable automata, register automata ...

incomparable!

Synthesis of Data Transducers



Synthesis problem:
given specification $S \subseteq (I \cdot O)^\omega$, return transducer with $L(T) \subseteq S$ or UNREAL

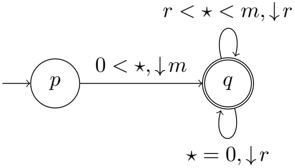
Finite-alphabet case:
spec: LTL, $<$ $\underbrace{\omega\text{-reg expressions, MSO, nondet/univ/det automata}}_{\text{equally expressive}}$

Data case:
spec: FO with $>_d$, Constraint LTL, LTL with freeze quantifier, pebble automata, variable automata, register automata ...

incomparable!

Register Automata on $(\mathbb{N}, <, 0)$

$$A = (Q, q_0, R, \delta, \alpha)$$



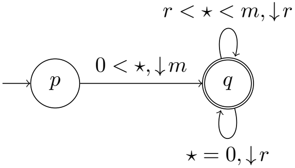
Run is a sequence of configurations.

Register Automata on $(\mathbb{N}, <, 0)$

states initial registers acceptance condition

$$A = (Q, q_0, R, \delta, \alpha)$$

Transition relation $\subseteq Q \times Tst \times Asgn \times Q$



Run is a sequence of configurations.

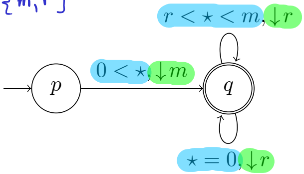
Register Automata on $(\mathbb{N}, <, 0)$

states — initial — registers — acceptance condition

$$A = (Q, q_0, R, \delta, \alpha)$$

Transition relation $\subseteq Q \times Tst \times Asgn \times Q$

$R = \{m, r\}$



Run is a sequence of configurations.

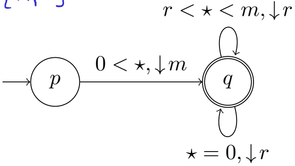
Register Automata on $(\mathbb{N}, <, 0)$

states · initial · registers · acceptance condition

$$A = (Q, q_0, R, \delta, \alpha)$$

Transition relation $\subseteq Q \times Tst \times Asgn \times Q$

$R = \{m, r\}$



$$(q, \underset{m}{0}, \underset{r}{0}) \longrightarrow (p, \underset{m}{5}, \underset{r}{0}) \longrightarrow (p, 5, 1) \longrightarrow (p, 5, 2) \rightarrow (p, 5, 4) \rightarrow (p, 5, 0)$$

Run is a sequence of configurations.

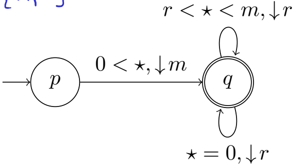
Register Automata on $(\mathbb{N}, <, 0)$

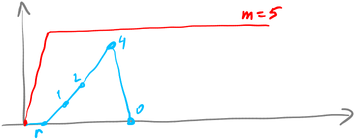
states, initial, registers, acceptance condition

$$A = (Q, q_0, R, \delta, \alpha)$$

Transition relation $\subseteq Q \times Tst \times Asgn \times Q$

$R = \{m, r\}$





$(q, \underset{m}{0}, \underset{r}{0}) \longrightarrow (p, \underset{m}{5}, \underset{r}{0}) \longrightarrow (p, 5, 1) \longrightarrow (p, 5, 2) \to (p, 5, 4) \to (p, 5, 0)$

Run is a sequence of configurations.

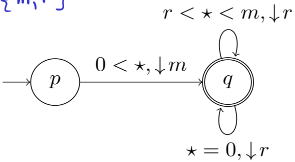
Register Automata on $(\mathbb{N}, <, 0)$

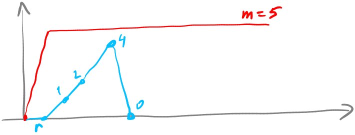
states, initial, registers, acceptance condition

$$A = (Q, q_0, R, \delta, \alpha)$$

Transition relation $\subseteq Q \times Tst \times Asgn \times Q$

$R = \{m, r\}$





$$(q, \underset{m}{0}, \underset{r}{0}) \longrightarrow (p, \underset{m}{5}, \underset{r}{0}) \longrightarrow (p, 5, 1) \longrightarrow (p, 5, 2) \rightarrow (p, 5, 4) \rightarrow (p, 5, 0)$$

Run is a sequence of configurations.

nondet universal det

variants

Data Domain

A *data domain* is a tuple $(\mathbb{D}, P, C, c_0)$, where:
$\mathbb{D}$ is a set of its elements,
P – interpreted predicates,
C – set of constants, c_0 is initialiser.

Data Domain

A *data domain* is a tuple $(\mathbb{D}, P, C, c_0)$, where:
$\mathbb{D}$ is a set of its elements,
P – interpreted predicates,
C – set of constants, c_0 is initialiser.

$(\mathbb{N}, \{<\}, \{0\}, 0)$

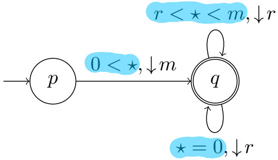
$(\mathbb{D}, \{=\}, \{0\}, 0)$

Data Domain

A *data domain* is a tuple $(\mathbb{D}, P, C, c_0)$, where:
$\mathbb{D}$ is a set of its elements,
P – interpreted predicates,
C – set of constants, c_0 is initialiser.

$(\mathbb{N}, \{<\}, \{0\}, 0)$

$(\mathbb{D}, \{=\}, \{0\}, 0)$



Tests are conjunctions of literals

Action words (tst, asgn) (tst, asgn) ...

FEAS – feasible action words

Synthesis of Register Transducers

Synthesis problem:
given: universal register automaton S
return: register transducer T with $L(T) \subseteq L(S)$

Synthesis of Register Transducers

Synthesis problem:
given: universal register automaton S
return: register transducer T with $L(T) \subseteq L(S)$

	$(\mathbb{D}, =)$	$(\mathbb{Q}, <)$	$(\mathbb{N}, <)$
unconstrained	✗ [2]	✗	✗

Synthesis of Register Transducers

Synthesis problem:
given: universal register automaton S
return: register transducer T with $L(T) \subseteq L(S)$

	$(\mathbb{D}, =)$	$(\mathbb{Q}, <)$	$(\mathbb{N}, <)$
unconstrained	✗[2]	✗	✗
reg-bounded	✓[1]	✓[2]	✓[3]

Register-bounded version:
given: universal register automaton S, *bound k*
return: k-register transducer T with $L(T) \subseteq L(S)$

[1]: R.Bloem, B.Maderbacher, A.K.: Bounded Synthesis of Register Transducers

[2]: L.Exibard: Automatic Synthesis of Systems with Data

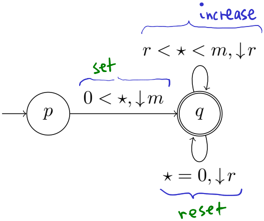
[3]: L.Exibard, E.Filiot, A.K.: Generic Solution to Register-bounded Synthesis

Recent Advances

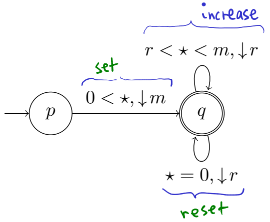
Reg-bounded synthesis is decidable
- for $(\mathbb{N}, <)$
- for *regapprox* domains

Reducibility between data domains.

Insight 1: Abstraction

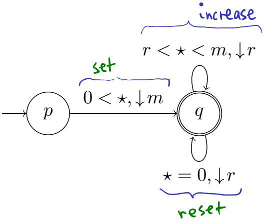


Insight 1: Abstraction



set $(\text{increase})^{\omega}$

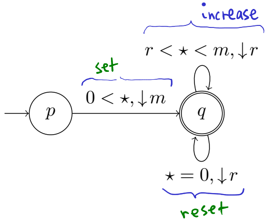
Insight 1: Abstraction



set $(\text{increase})^\omega$

set incr reset incr^2 reset incr^3 reset ...

Insight 1: Abstraction



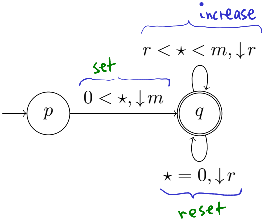
set $(\text{increase})^{\omega}$

set incr reset incr^2 reset incr^3 reset ...

Any feasible action word has the form

set incr^{n_1} reset incr^{n_2} ... where $\exists B$: every $n_i < B$

Insight 1: Abstraction



set $(\text{increase})^\omega$

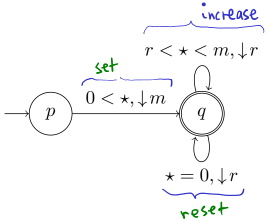
set incr reset incr^2 reset incr^3 reset ...

Any feasible action word has the form

set incr^{n_1} reset incr^{n_2} ... where $\exists B$: every $n_i < B$

An *action word* is a sequence $(\mathsf{tst}_0, \mathsf{asgn}_0)(\mathsf{tst}_1, \mathsf{asgn}_1)\ldots$.
It is *feasible* if it is induced by some data word.

Insight 1: Abstraction



set (increase)$^\omega$

set incr reset incr^2 reset incr^3 reset ...

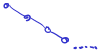
Any feasible action word has the form

set incr^{n_1} reset incr^{n_2} ... where $\exists B$: every $n_i < B$

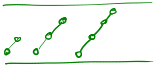
An *action word* is a sequence $(\mathsf{tst}_0, \mathsf{asgn}_0)(\mathsf{tst}_1, \mathsf{asgn}_1)\dots$.
It is *feasible* if it is induced by some data word.

In $(\mathbb{N}, <)$, action word is feasible iff it has:

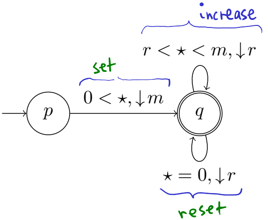
no inf decreasing chains



no unbounded chains of the form



Insight 1: Abstraction



set (increase)$^\omega$

set incr reset incr^2 reset incr^3 reset ...

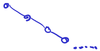
Any feasible action word has the form

set incr^{n_1} reset incr^{n_2} ... where $\exists B$: every $n_i < B$

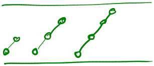
An *action word* is a sequence $(\mathsf{tst}_0, \mathsf{asgn}_0)(\mathsf{tst}_1, \mathsf{asgn}_1)\dots$.
It is *feasible* if it is induced by some data word.

In $(\mathbb{N}, <)$, action word is feasible iff it has:

no inf decreasing chains



no unbounded chains of the form



Let FEAS be the set of feasible action words over given R.

Insight 1: Abstraction

Given S and k, create a *finite-alphabet* specification $W_{S,k}$:

$W_{S,k}$ is realizable by a Mealy machine

$\Leftrightarrow$

S is realizable by a k-reg transducer.

Insight 1: Abstraction

Given S and k, create a *finite-alphabet* specification $W_{S,k}$:

$W_{S,k}$ is realizable by a Mealy machine

$\Leftrightarrow$

S is realizable by a k-reg transducer.

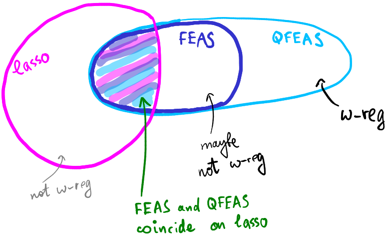
$$W^{\mathsf{F}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\big\}.$$

Solving such a synthesis problem is hard, as FEAS is not ω-regular :-(

Generic Solution (Insight 2)

Data domain is *regapprox* if for every R there exists eff.constr. ω-regular over-approximation QFEAS

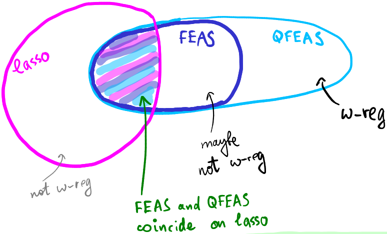
$$\mathsf{QFEAS} \cap lasso \subseteq \mathsf{FEAS} \subseteq \mathsf{QFEAS}.$$



Generic Solution

Data domain is *regapprox* if for every R there exists eff.constr. ω-regular over-approximation QFEAS

$$\mathsf{QFEAS} \cap lasso \subseteq \mathsf{FEAS} \subseteq \mathsf{QFEAS}.$$



Theorem:
on regapprox domains, register-bounded synthesis is decidable.

Proof Idea

L1: S is realisable by k-reg transducer iff $W^{\mathsf{F}}_{S,k}$ is realisable:

$$W^{\mathsf{F}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\big\}.$$

L2: $W^{\mathsf{F}}_{S,k}$ is realisable iff $W^{\mathsf{QF}}_{S,k}$ is realisable:

$$W^{\mathsf{QF}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{QFEAS}\big\}.$$

Proof Idea

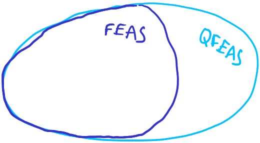
L1: S is realisable by k-reg transducer iff $W^{\mathsf{F}}_{S,k}$ is realisable:

$$W^{\mathsf{F}}_{S,k} = \neg\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\}.$$

L2: $W^{\mathsf{F}}_{S,k}$ is realisable iff $W^{\mathsf{QF}}_{S,k}$ is realisable:

$$W^{\mathsf{QF}}_{S,k} = \neg\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{QFEAS}\}.$$

Proof idea. W^{QF} is realisable $\Longrightarrow$ W^{F} is realisable



Proof Idea

L1: S is realisable by k-reg transducer iff $W^{\mathsf{F}}_{S,k}$ is realisable:

$$W^{\mathsf{F}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\big\}.$$

L2: $W^{\mathsf{F}}_{S,k}$ is realisable iff $W^{\mathsf{QF}}_{S,k}$ is realisable:

$$W^{\mathsf{QF}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{QFEAS}\big\}.$$

Proof idea. W^{QF} is realisable $\Longleftarrow$ W^{F} is realisable

Proof Idea

L1: S is realisable by k-reg transducer iff $W^{\mathsf{F}}_{S,k}$ is realisable:

$$W^{\mathsf{F}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\big\}.$$

L2: $W^{\mathsf{F}}_{S,k}$ is realisable iff $W^{\mathsf{QF}}_{S,k}$ is realisable:

$$W^{\mathsf{QF}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{QFEAS}\big\}.$$

Proof idea. W^{QF} is realisable $\Longleftarrow$ W^{F} is realisable

By contradiction. Suppose T realises but not W^{QF}.

Show that $T \not\models W^{QF} \implies T \not\models W^{F}$:

Proof Idea

L1: S is realisable by k-reg transducer iff $W^{\mathsf{F}}_{S,k}$ is realisable:

$$W^{\mathsf{F}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\big\}.$$

L2: $W^{\mathsf{F}}_{S,k}$ is realisable iff $W^{\mathsf{QF}}_{S,k}$ is realisable:

$$W^{\mathsf{QF}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{QFEAS}\big\}.$$

Proof idea. W^{QF} is realisable $\Leftarrow$ W^{F} is realisable

By contradiction. Suppose T realises but not W^{QF}.

Show that $T \not\models W^{QF} \Longrightarrow T \not\models W^{F}$:

$$\underbrace{T \cap \overline{W^{QF}}}_{\omega\text{-reg}} \neq \emptyset$$

Proof Idea

L1: S is realisable by k-reg transducer iff $W^{\mathsf{F}}_{S,k}$ is realisable:

$$W^{\mathsf{F}}_{S,k} = \neg\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\}.$$

L2: $W^{\mathsf{F}}_{S,k}$ is realisable iff $W^{\mathsf{QF}}_{S,k}$ is realisable:

$$W^{\mathsf{QF}}_{S,k} = \neg\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{QFEAS}\}.$$

Proof idea. W^{QF} is realisable $\Longleftarrow$ W^{F} is realisable

By contradiction. Suppose T realises but not W^{QF}.

Show that $T \not\models W^{QF} \Longrightarrow T \not\models W^{F}$:

$\underbrace{T \cap \overline{W^{QF}}}_{\omega\text{-reg}} \neq \emptyset \implies \exists$ lasso: $\bar{a}_T \otimes \bar{a}_S \in QFEAS$

$\bar{a}_S \in L(\overline{S_{synt}})$

Proof Idea

L1: S is realisable by k-reg transducer iff $W^{\mathsf{F}}_{S,k}$ is realisable:

$$W^{\mathsf{F}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{FEAS}\big\}.$$

L2: $W^{\mathsf{F}}_{S,k}$ is realisable iff $W^{\mathsf{QF}}_{S,k}$ is realisable:

$$W^{\mathsf{QF}}_{S,k} = \neg\big\{\bar{a}_T \mid \exists \bar{a}_S \in L(\overline{S_{synt}})\colon \bar{a}_T \otimes \bar{a}_S \in \mathsf{QFEAS}\big\}.$$

Proof idea. W^{QF} is realisable $\Longleftarrow$ W^F is realisable

By contradiction. Suppose T realises but not W^{QF}.

Show that $T \not\models W^{QF} \implies T \not\models W^F$:

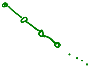
$\underbrace{T \cap \overline{W^{QF}}}_{\omega\text{-reg}} \neq \emptyset \implies \exists$ lasso: $\bar{a}_T \otimes \bar{a}_S \in QFEAS \implies \bar{a}_T \otimes \bar{a}_S \in FEAS$

$\in L(\overline{S_{synt}})$

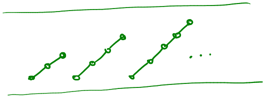
$\bar{a}_T \in \overline{W^F}$ ↯

DOMAIN $(\mathbb{N}, <)$ is regapprox!

Recall that in $(\mathbb{N}, <)$ action word is feasible iff there are no:



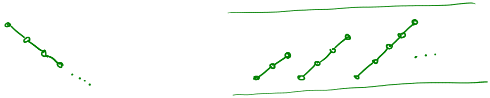
inf decr chains



unbounded chains

DOMAIN $(\mathbb{N}, <)$ is regapprox!

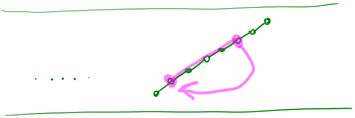
Recall that in $(\mathbb{N}, <)$ action word is feasible iff there are no:



inf decr chains

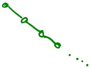
unbounded chains

When considering lasso words:

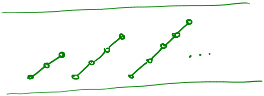


Domain $(\mathbb{N}, <)$ is regapprox!

Recall that in $(\mathbb{N}, <)$ action word is feasible iff there are no:

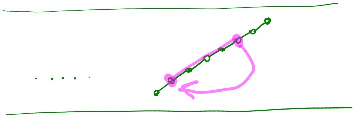


inf decr chains

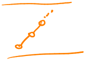


unbounded chains

When considering lasso words:



$\Rightarrow$ enough to check absence of inf chains



Main Theorem

Reg-bounded synthesis in $(\mathbb{N}, <)$ is solvable in time
$$exp(exp(r, k), n, c)$$
for every given universal parity register automaton with r registers, n states, c priorities, and bound k.

A similar complexity holds for domains $(\mathbb{Q}, <)$ and $(\mathbb{D}, =)$.

Reduction between Domains

If $\mathcal{D}$ reduces *to* $\mathcal{D}'$*, and* $\mathcal{D}'$ *is regapprox, then* D *is regapprox.*

Two definitions of reductions:
- via transducer relations,
- via first-order formulas.

action word in D ⇝ action words in D'
feas ⟹ ∃ feas

Allows us to state decidability of register-bounded synthesis for $(\mathbb{N}^d, <^d)$ and $(\Sigma^*, \prec)$.

Conclusion

	$(\mathbb{D}, =)$	$(\mathbb{Q}, <)$	$(\mathbb{N}, <)$
unconstrained	✗[2]	✗	✗
reg-bounded	✓[1]	✓[2]	✓[3]

+ generic result

Conclusion

	$(\mathbb{D}, =)$	$(\mathbb{Q}, <)$	$(\mathbb{N}, <)$
unconstrained	✗ [2]	✗	✗
reg-bounded	✓[1]	✓[2]	✓[3]

+ generic result

Experiments (fresh)

TO = 3600 seconds			
	translation time (Q,<) | (D,=)	# states (Q,<) | (D,=)	synthesis time (Q,<) | (D,=)
buffer 1	**0 | 0**	**40 | 16**	**0 | 0**
buffer 2	**1 | 0**	**301 | 61**	**10 | 1**
buffer 3	**36 | 0**	**2706 | 261**	**TO | 17**
buffer 4	**1241 | 11**	**28099 | 1219**	**TO | TO**
buffer 5	**TO | 164**	**-- | 6140**	
buffer 6	**TO | 2497**	**-- | 33121**	
buffer 7	**TO | TO**		

Story of Register-bounded Synthesis

2014: R.Ehlers, S.Seshia, H.Kress-Gazit:
Synthesis with Identifiers

2018: A.K., B.Maderbacher, R.Bloem:
Bounded Synthesis of Register Transducers

2019: A.K., O.Kupferman:
Register-bounded Synthesis

L.Exibard, E.Filiot, P-A.Reynier:
Synthesis of Data Word Transducers

2021: L.Exibard, E.Filiot, A.K.:
Church Synthesis on Register Automata over Linearly Ordered Data Domains

2022: L.Exibard, E.Filiot, A.K.:
Generic Solution to Register-bounded Synthesis