Universality for Timed Automata with Minimal Resources

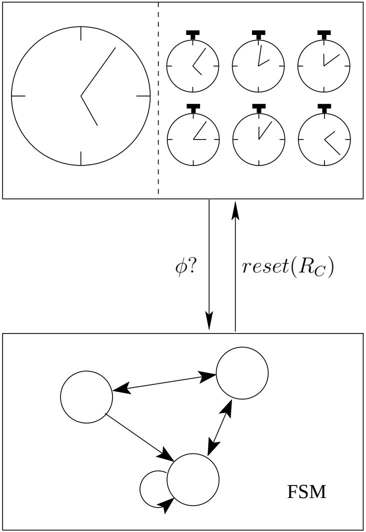
Sara E. Adams

in collaboration with
Joël Ouaknine and James Worrell

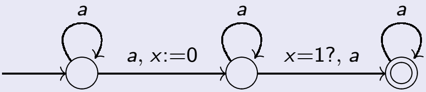
University of Oxford

FORMATS 2007

Concept of a timed automaton



An example timed automaton



Motivation

Why universality?

Special case of language inclusion

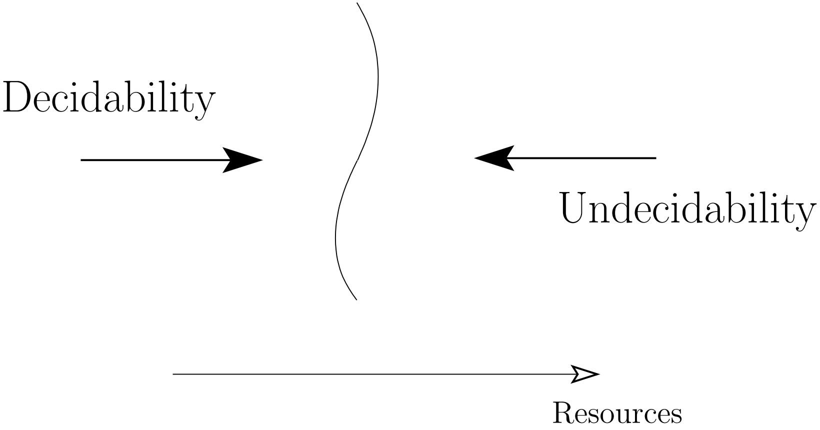
- All timed words $\subseteq$ L $\Rightarrow$ L universal
- Universality undecidable $\Rightarrow$ language inclusion undecidable

Why language inclusion?

Verification of real-time systems

- Essential role of language inclusion
- e.g. "Implementation $\subseteq$ Specification"

Motivation



Universality Problem

Does a given automaton accept every timed word?

Alur and Dill, 1994

Universality is undecidable for timed automata with two clocks.

Ouaknine and Worrell, 2004

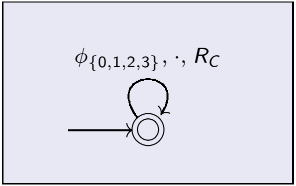
Universality is decidable for timed automata with one clock.

Our contribution

Adams, Ouaknine, Worrell

Universality is undecidable for timed automata with one state, one event and comparisons to

- weakly monotonic: 0 and 1 only.
- strongly monotonic: 1, 2, and 3 only.



Minimal resources

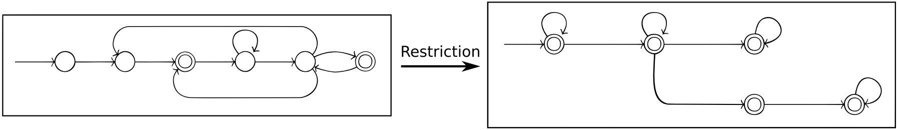
Obviously minimal

- essentially stateless
- essentially alphabetless

Further observations

- comparisons to 0 only: decidable
- restrictions on number of clocks: decidable

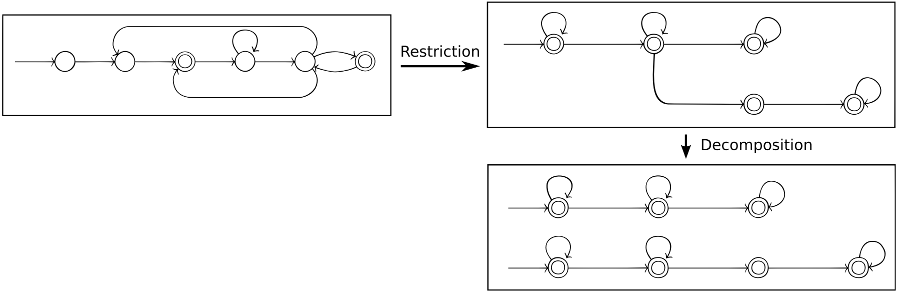
Structure of the proof



Basic steps

1. Universality for linear safety automata

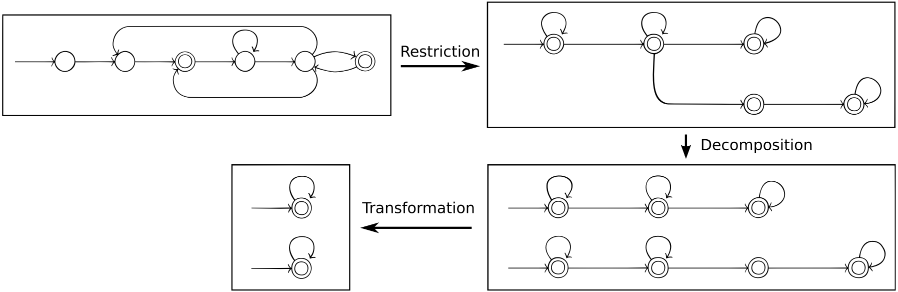
Structure of the proof



Basic steps

1. Universality for linear safety automata
2. Decomposition of linear safety automata

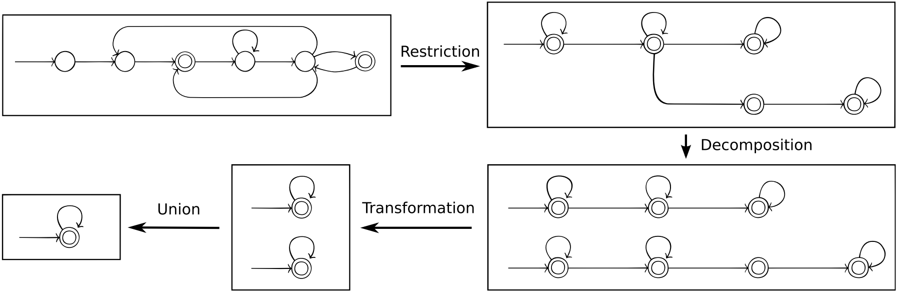
Structure of the proof



Basic steps

1. Universality for linear safety automata
2. Decomposition of linear safety automata
3. Transformation of decomposed timed automata

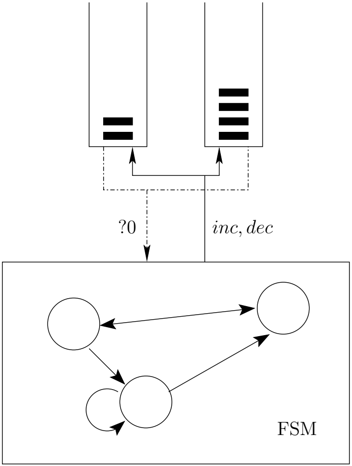
Structure of the proof



Basic steps

1. Universality for linear safety automata
2. Decomposition of linear safety automata
3. Transformation of decomposed timed automata
4. Union of transformed automata

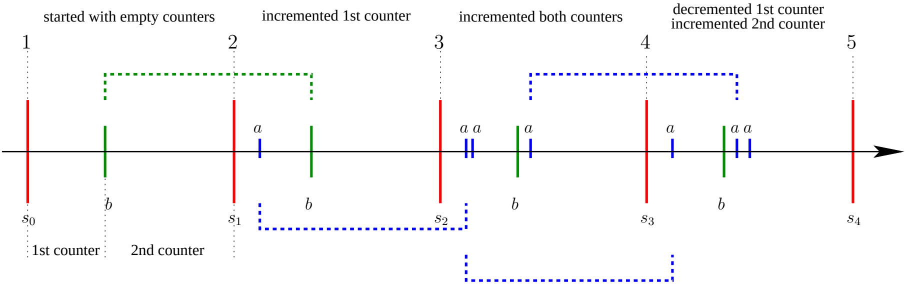
Universality Proof: 2-counter machines



Weakly monotonic time: 2-counter machine encoding

Task

- Accept any inconsistent encoding of a halting computation
- Automaton universal ⇔ 2-counter machine does not halt



Weakly monotonic time: Converting to one symbol

Simultaneous events

Encode alphabet symbols with simultaneous events

Change of time model

before: strongly monotonic encoding
after: weakly monotonic encoding

Weakly monotonic time: Converting to one state

Clocks

For every state introduce a seperate clock

Rule

Use predicates to encode states:

- Reset state clock on state transition

Essentials

- Linearity of automata
- On inconsistency: predicate that ensures acceptance

Strongly monotonic time: 2-counter machine encoding

Problem

Cannot use simultaneous events to encode alphabet

Solution

Use only one symbol in the 2-counter machine encoding

Price

Use 3 time units for each configuration:

- 1st time unit: encode the state
- 2nd time unit: encode the 1st counter
- 3rd time unit: encode the 2nd counter

Summing up

Universality is undecidable for timed automata with

- a single state,
- a single event, and
- clock comparisons to
  - weakly monotonic: 0 and 1 only.
  - strongly monotonic: 1, 2, and 3 only.

Open questions

Weakly monotonic time

Is universality undecidable for timed automata with:
one state, one event, and comparisons to 1 only?

Strongly monotonic time

Is universality undecidable for timed automata with:
one state, one event, and comparisons to one or two constants only?