Automata with timed atoms

Sławomir Lasota
University of Warsaw

joint work with Mikołaj Bojańczyk and Lorenzo Clemente

Infinity 2015, Bengaluru

FO-definable automata

Sławomir Lasota
University of Warsaw

joint work with Mikołaj Bojańczyk and Lorenzo Clemente

Infinity 2015, Bengaluru

FO-definable sets

offer a right setting for timed models of computation, like timed automata, or timed pushdown automata.

Plan

Plan

- Motivation

Plan

- Motivation
- FO-definable NFA

Plan

- Motivation
- FO-definable NFA
- FO-definable PDA

Plan

- Motivation
- FO-definable NFA
- FO-definable PDA
- The core problem: equations over sets of integers

Time domain

- reals
- rationals
- integers

dense time

discrete time

any choice of time domain is fine

Time domain

- reals
- rationals
- integers

dense time

discrete time

any choice of time domain is fine

Time domain

- reals
- rationals
- integers

dense time

discrete time

any choice of time domain is fine

No restriction to non-negative!

Time domain

- reals
- rationals
- integers

dense time

discrete time

any choice of time domain is fine

No restriction to non-negative!

Let input alphabet be reals

Time domain

- reals
- rationals
- integers

dense time

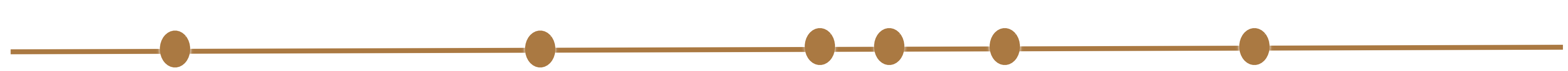
discrete time

any choice of time domain is fine

No restriction to non-negative!

Let input alphabet be reals

Monotonic input words :



Timed automata [Alur, Dill 1990]
with uninitialized clocks

Timed automata [Alur, Dill 1990]

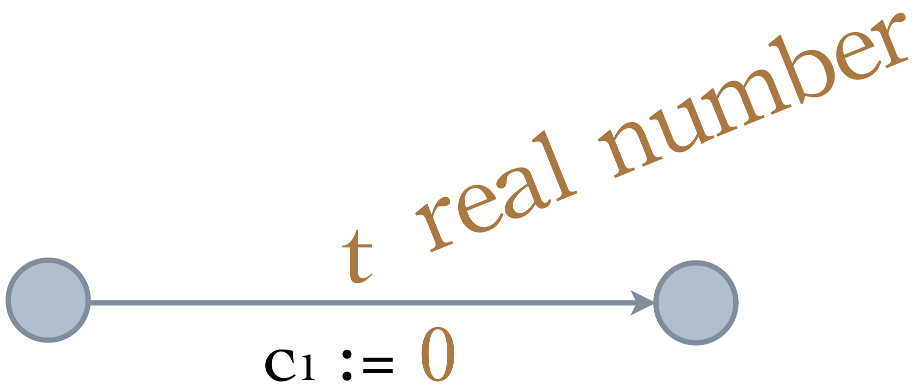
with uninitialized clocks

?

Timed automata [Alur, Dill 1990]
with uninitialized clocks

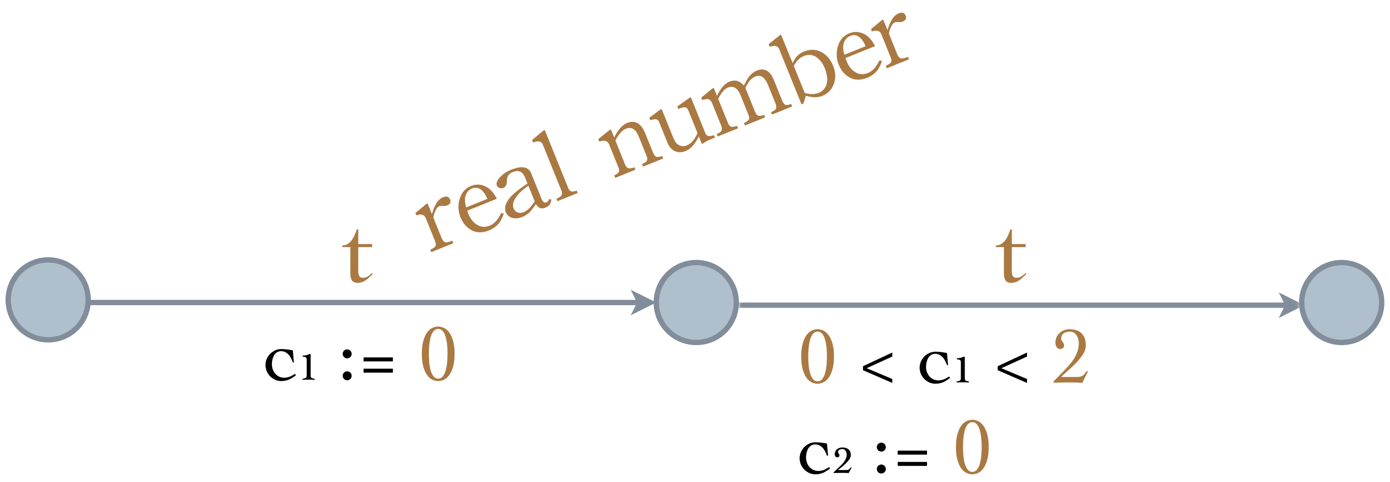
Timed automata [Alur, Dill 1990]

with uninitialized clocks



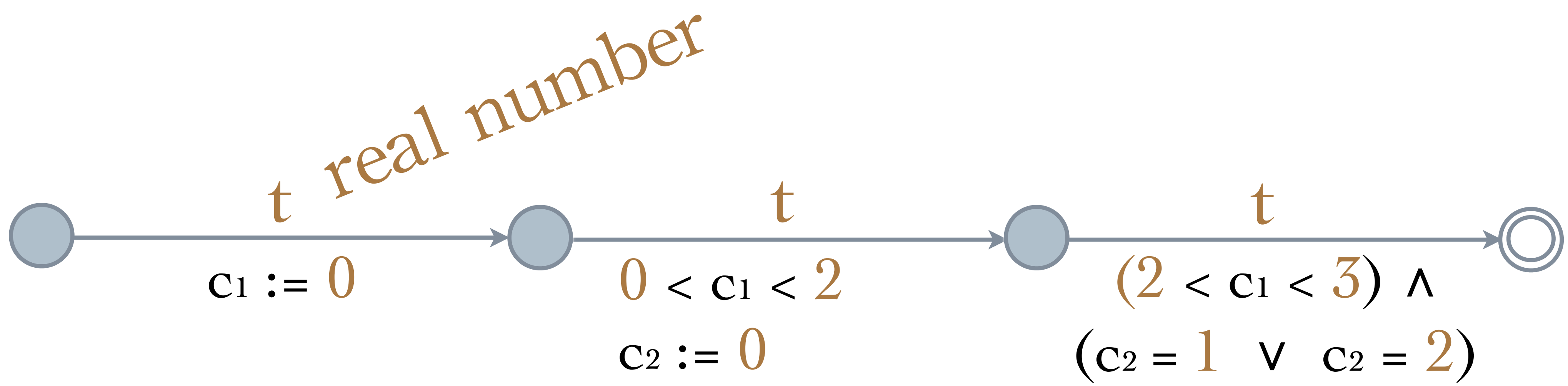
Timed automata [Alur, Dill 1990]

with uninitialized clocks



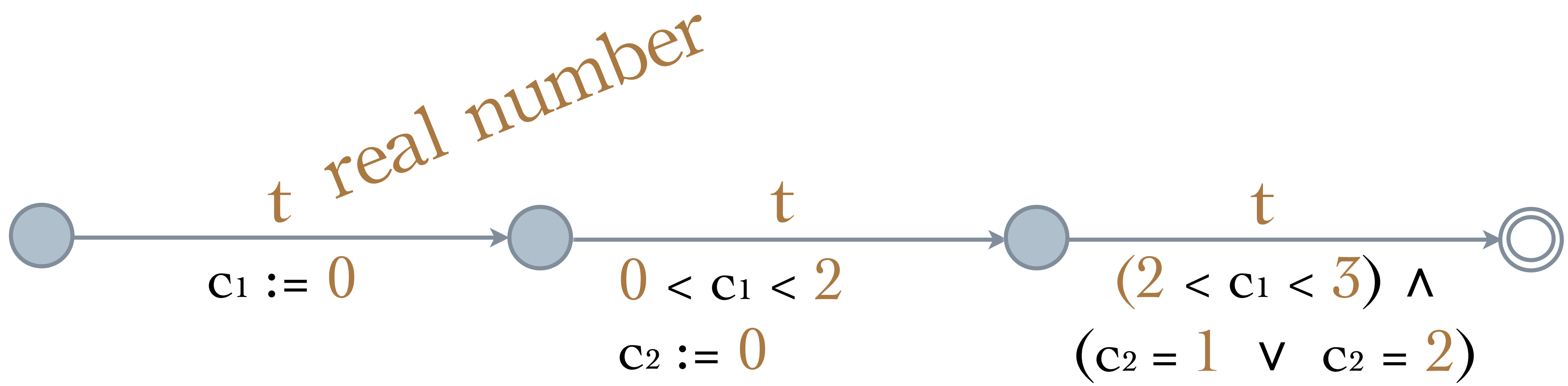
Timed automata [Alur, Dill 1990]

with uninitialized clocks

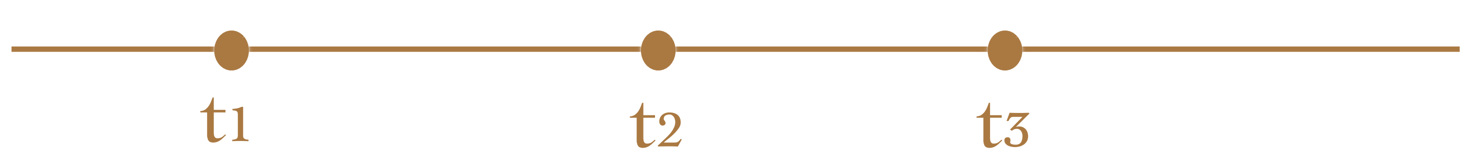


Timed automata [Alur, Dill 1990]

with uninitialized clocks

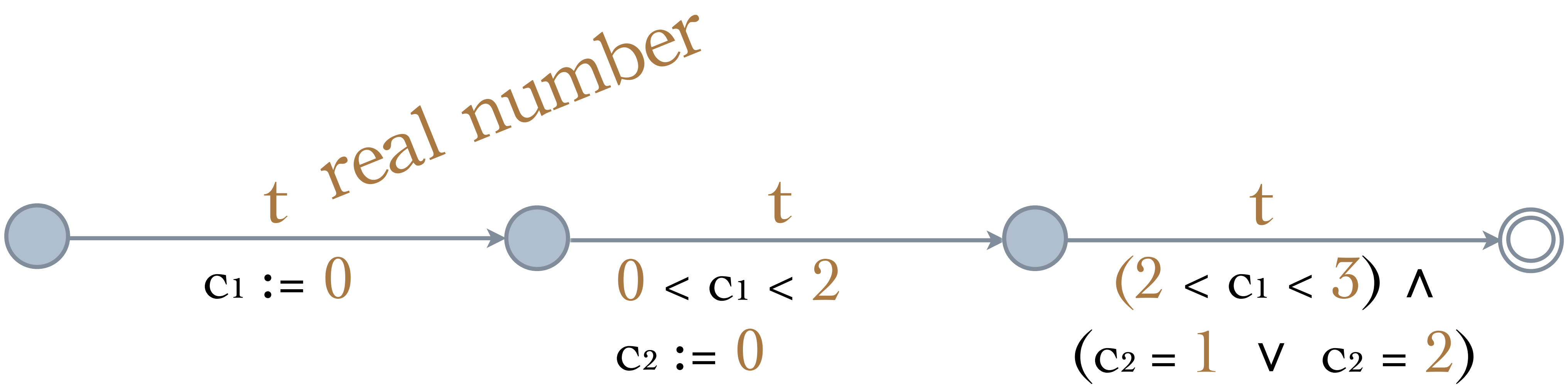


the automaton accepts words $t_1\ t_2\ t_3 \in \mathbb{R}^3$ such that

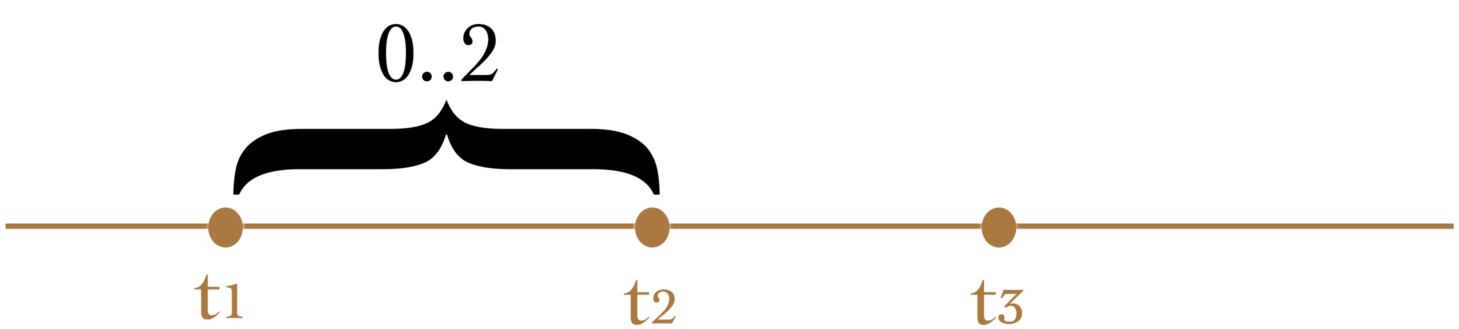


Timed automata [Alur, Dill 1990]

with uninitialized clocks

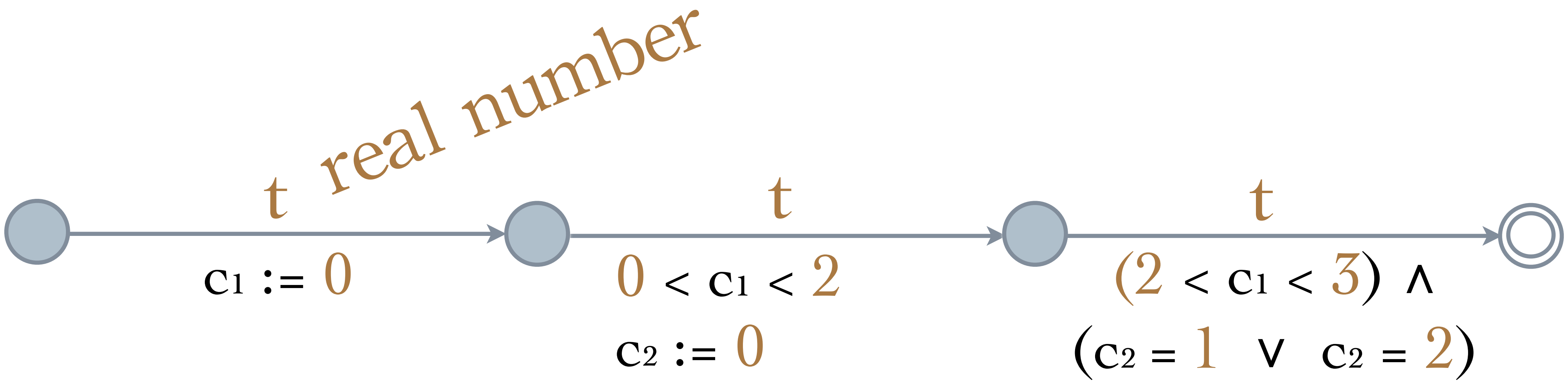


the automaton accepts words $t_1\ t_2\ t_3 \in R^3$ such that

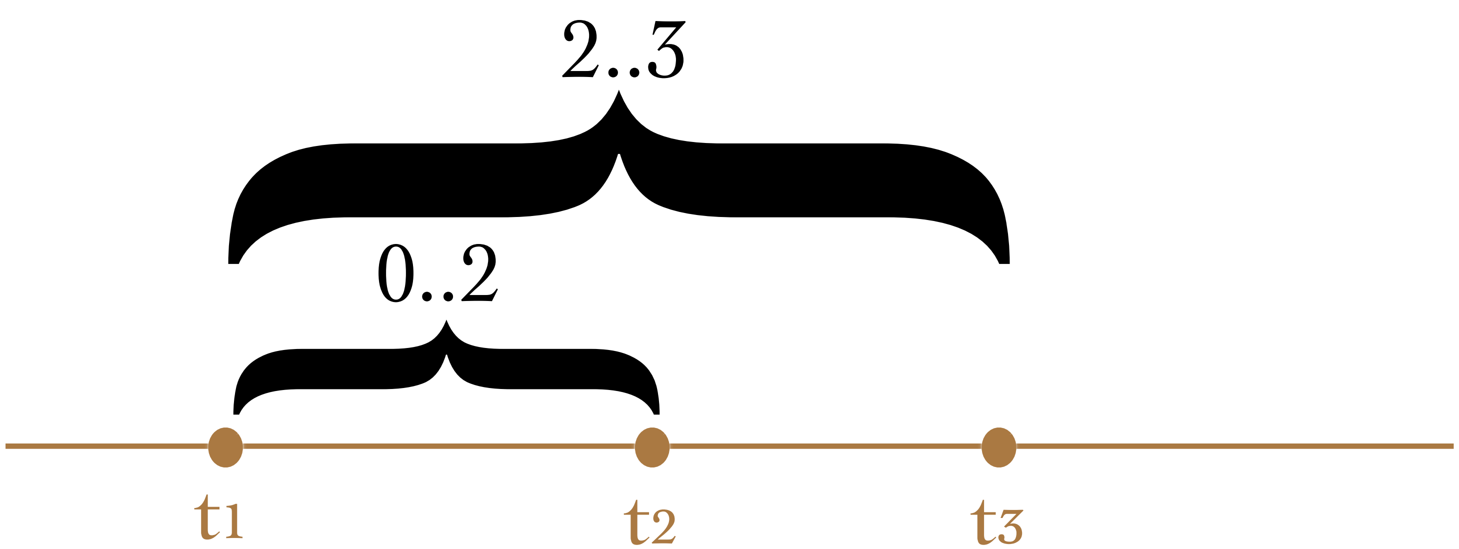


Timed automata [Alur, Dill 1990]

with uninitialized clocks

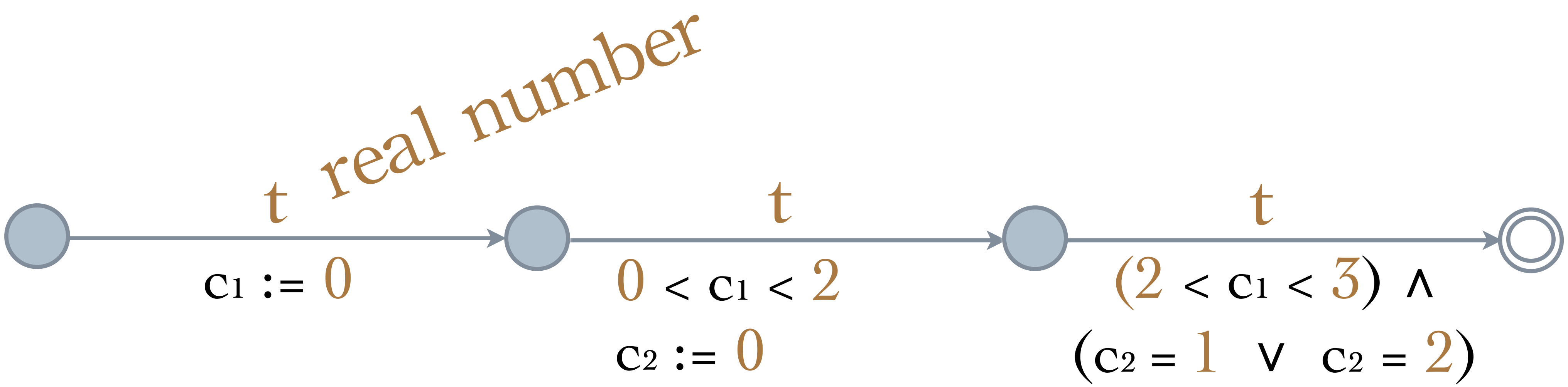


the automaton accepts words $t_1\ t_2\ t_3 \in \mathbb{R}^3$ such that

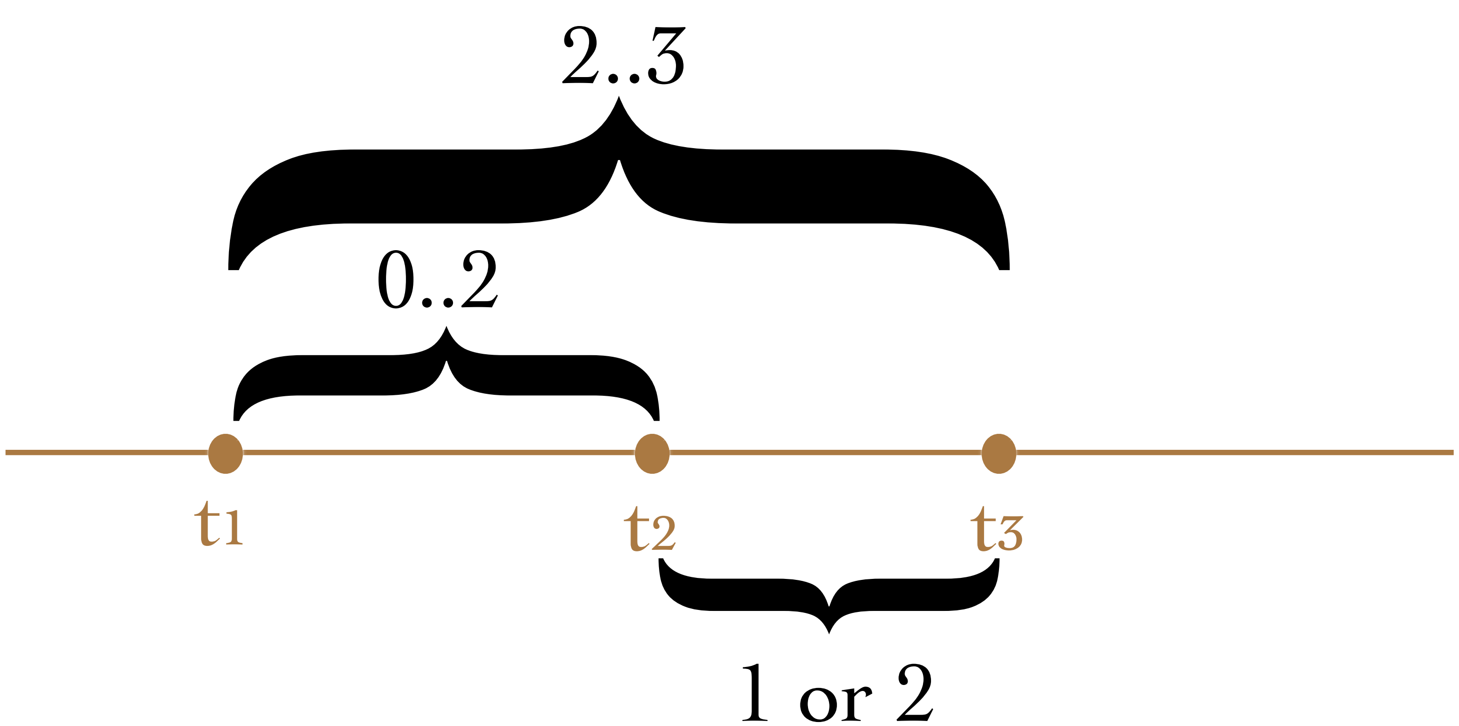


Timed automata [Alur, Dill 1990]

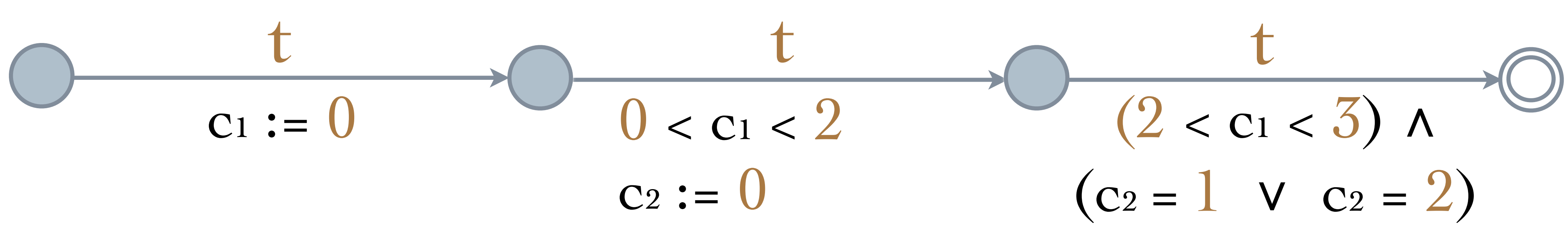
with uninitialized clocks

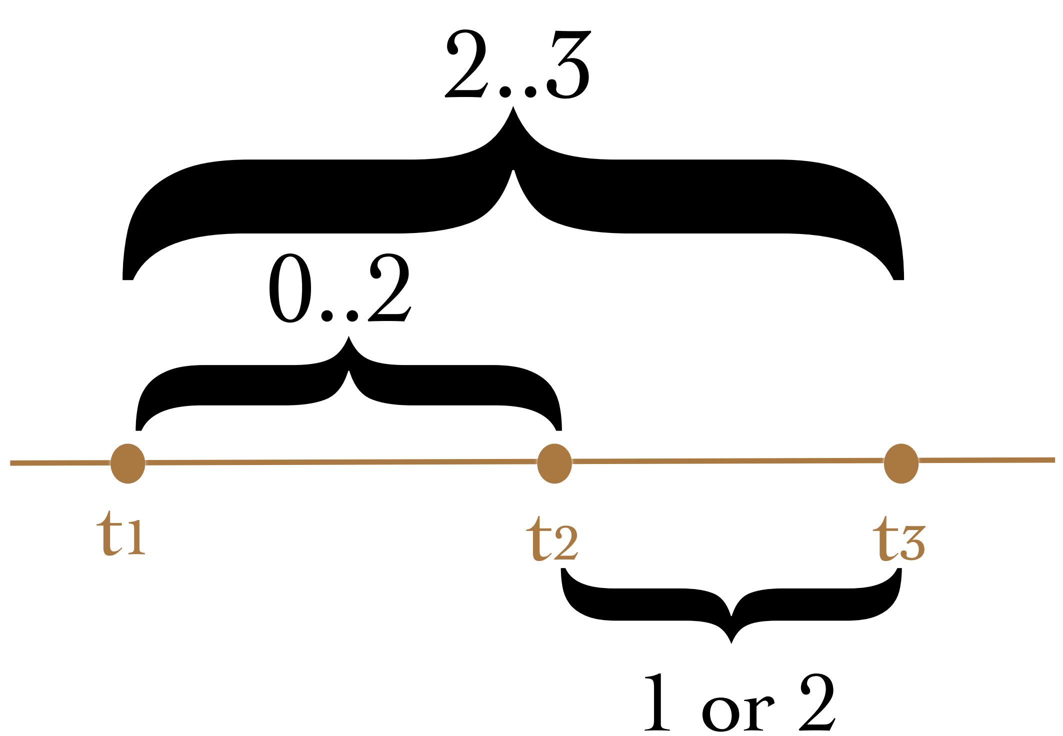


the automaton accepts words $t_1\ t_2\ t_3 \in \mathbb{R}^3$ such that

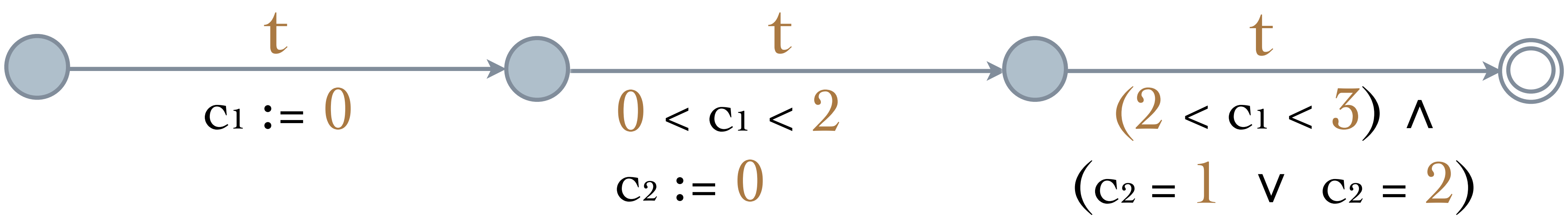


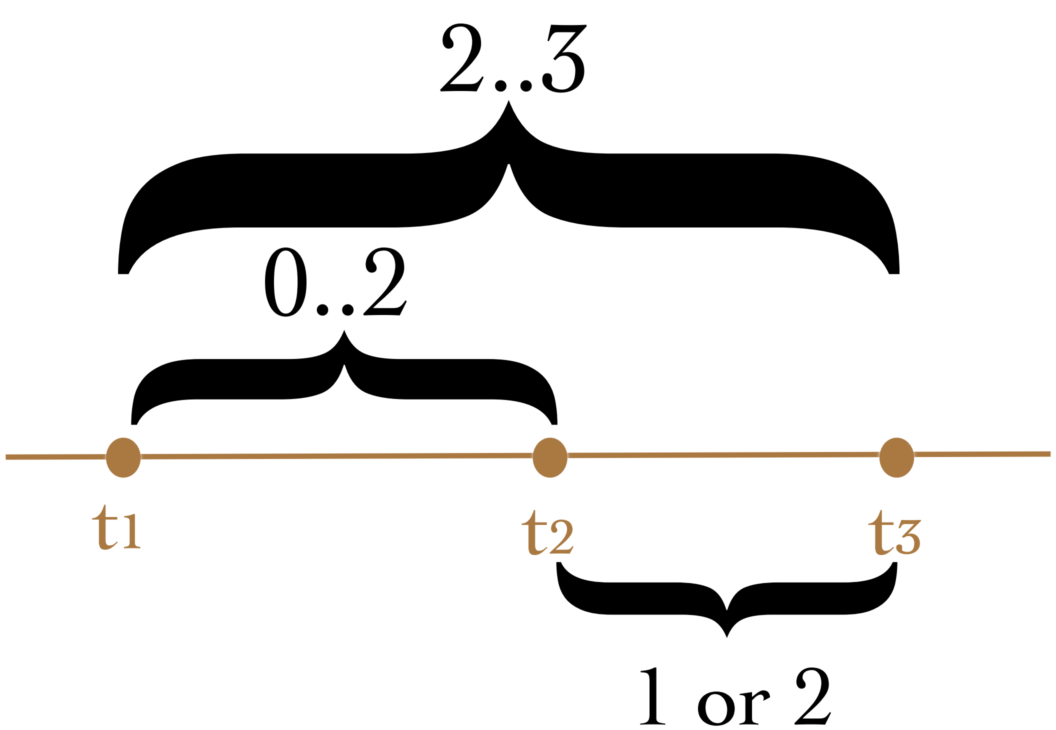
Deterministic timed automata don't minimize

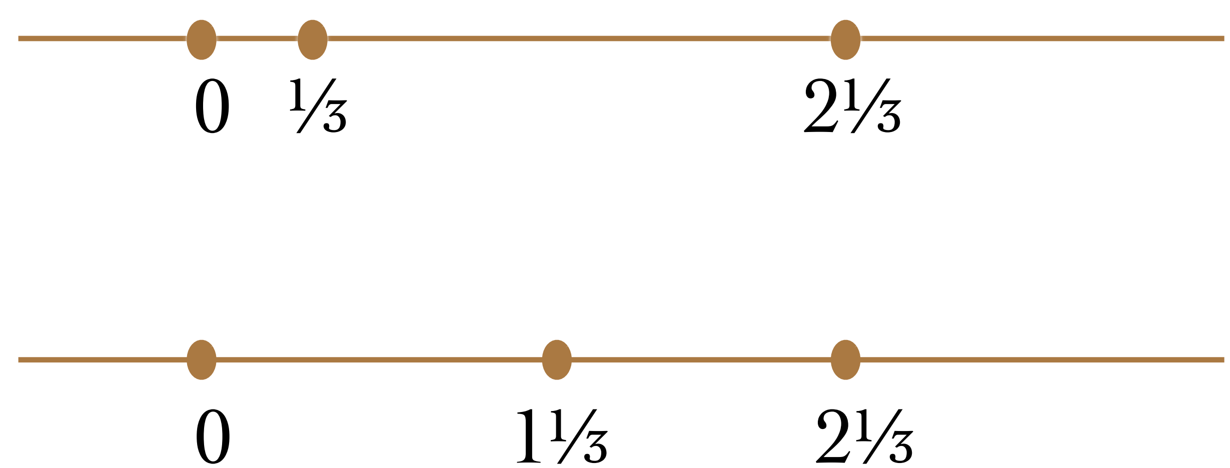




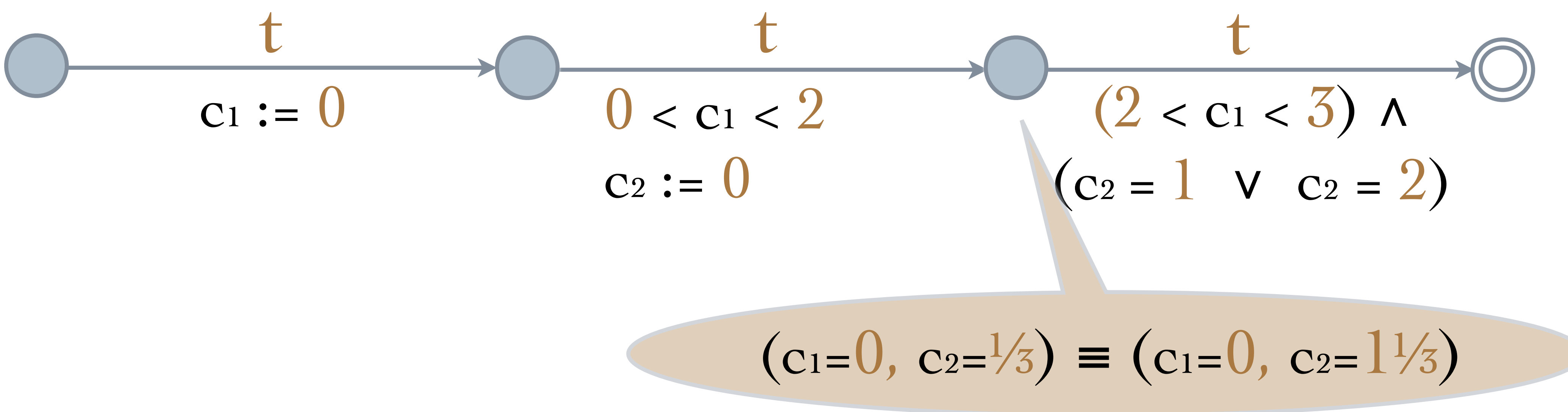
Deterministic timed automata don't minimize

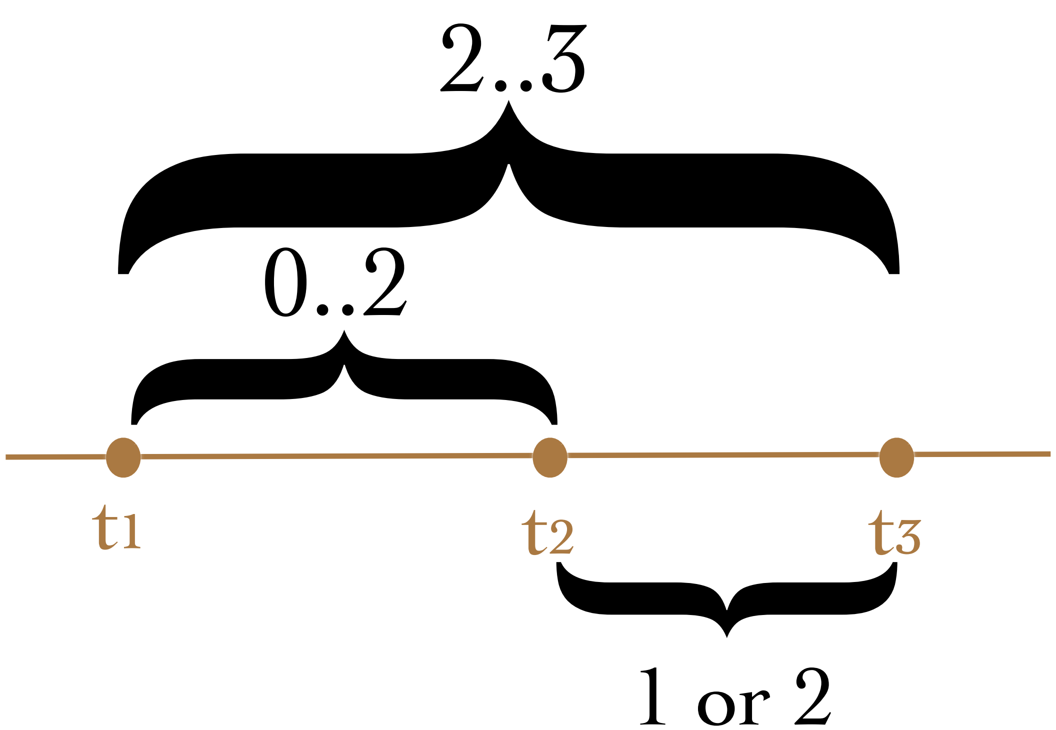


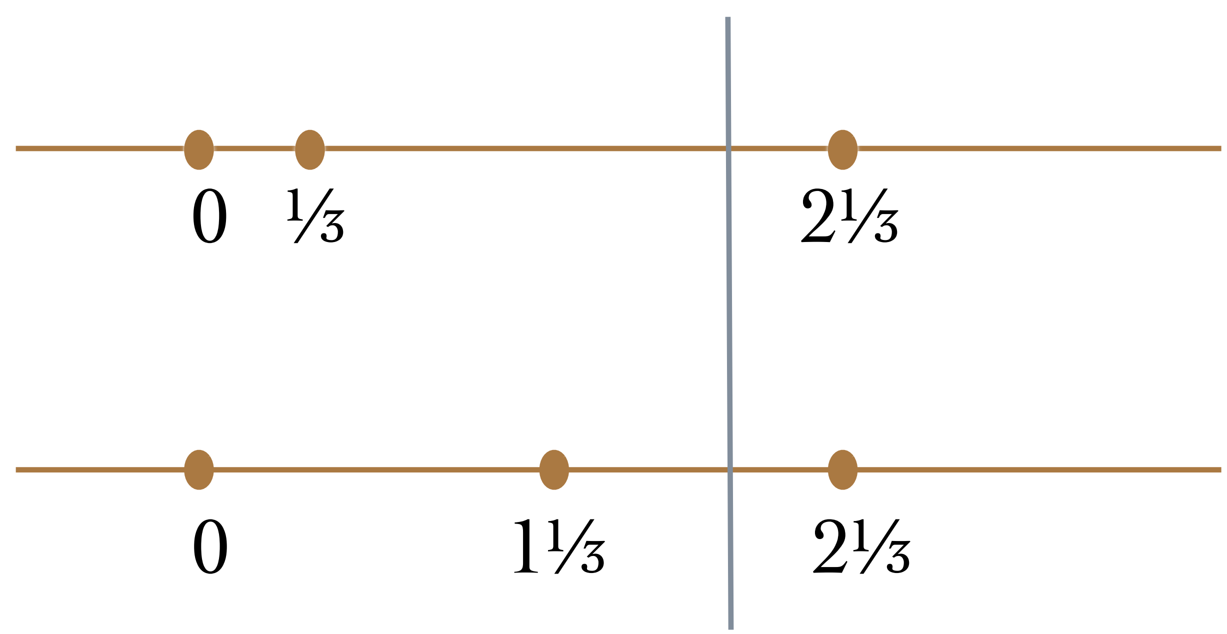




Deterministic timed automata don't minimize







Towards timed pushdown automata

Towards timed pushdown automata

- timed automata [Alur, Dill 1990]

Towards timed pushdown automata

- timed automata [Alur, Dill 1990]
- pushdown timed automata [Bouajjani, Echahed, Robbana 1994]

finite stack alphabet

Towards timed pushdown automata

- timed automata [Alur, Dill 1990]
- pushdown timed automata [Bouajjani, Echahed, Robbana 1994]
- dense-timed pushdown automata [Abdulla, Atig, Stenman 2012]

finite stack alphabet

- clocks can be pushed onto stack
- the emptiness problem EXPTIME-complete

Towards timed pushdown automata

- timed automata [Alur, Dill 1990]
- pushdown timed automata [Bouajjani, Echahed, Robbana 1994]
- dense-timed pushdown automata [Abdulla, Atig, Stenman 2012]
- recursive timed automata
  [Trivedi, Wojtczak 2010], [Benerecetti, Minopoli, Peron 2010]

finite stack alphabet

- clocks can be pushed onto stack
- the emptiness problem EXPTIME-complete

Dense-timed PDA collapse

Theorem 1: [Clemente, L. 2015]
Dense-timed pushdown automata are expressively equivalent to pushdown timed automata.

Dense-timed PDA collapse

Theorem 1: [Clemente, L. 2015]
Dense-timed pushdown automata are expressively equivalent to pushdown timed automata.

An accidental combination of

- stack discipline
- monotonicity of time
- syntactic restrictions

FO-definable sets

offer a right setting for timed models of computation, like timed automata, or timed pushdown automata.

FO-definable sets

offer a right setting for timed models of computation, like timed automata, or timed pushdown automata.

- do not invent a new definition

FO-definable sets

offer a right setting for timed models of computation, like timed automata, or timed pushdown automata.

- do not invent a new definition
- re-interpret a classical definition in FO-definable sets, with finiteness relaxed to **orbit-finiteness**

In search of lost definition

- Motivation
- FO-definable NFA
- FO-definable PDA
- The core problem: equations over sets of integers

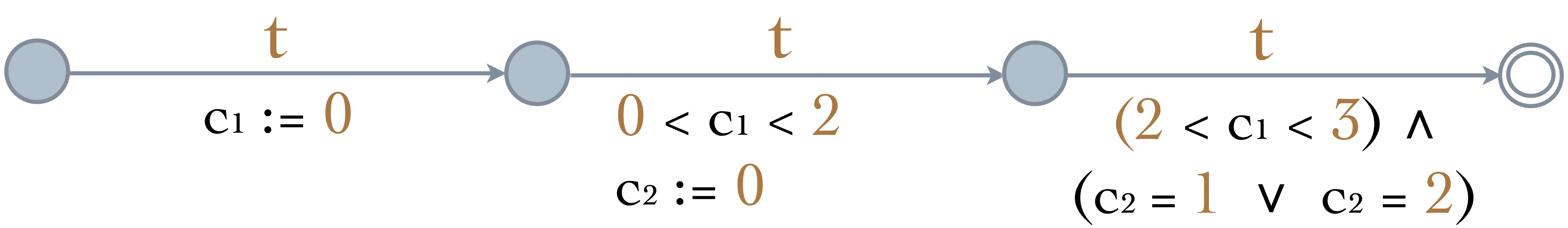
In search of lost definition

- Motivation
- FO-definable NFA
- FO-definable PDA
- The core problem: equations over sets of integers

NFA re-interpreted in FO-definable sets

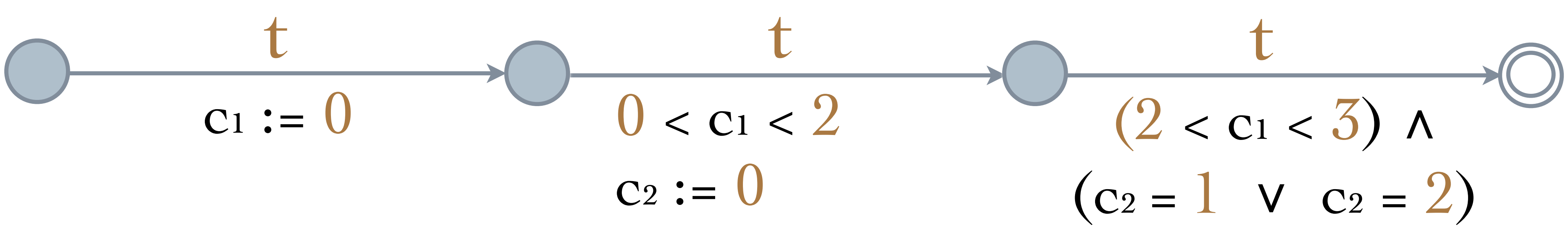
Timed automata are register automata

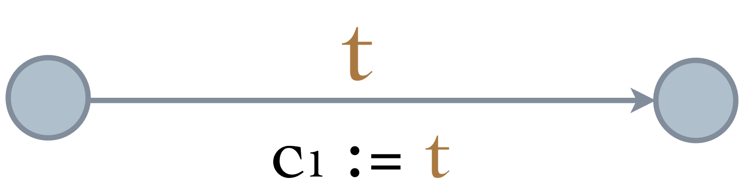
[Bojańczyk, L. 2012]



Timed automata are register automata

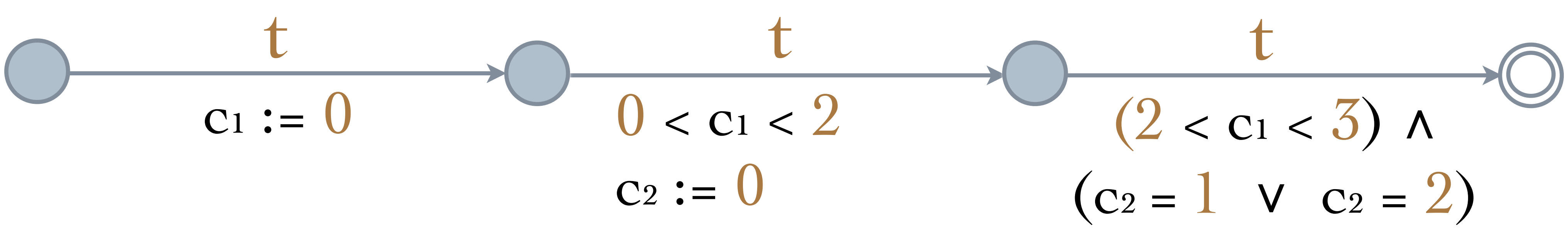
[Bojańczyk, L. 2012]

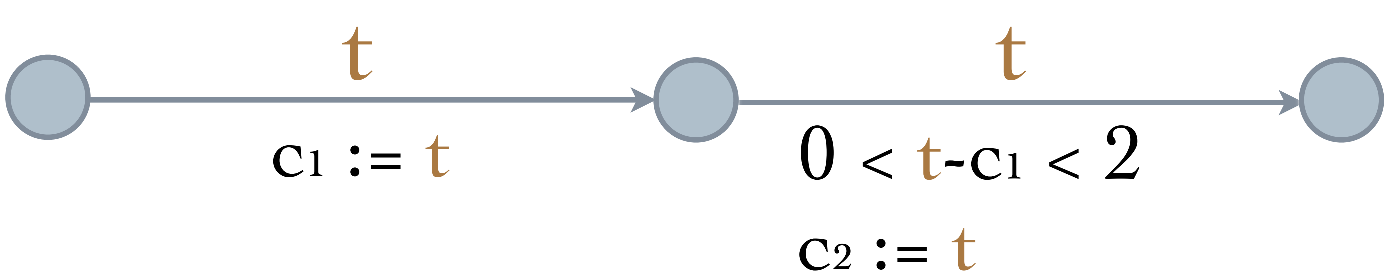




Timed automata are register automata

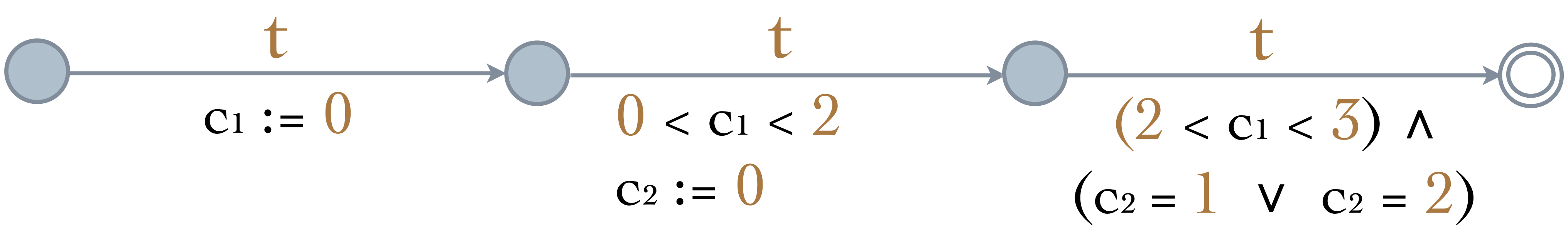
[Bojańczyk, L. 2012]

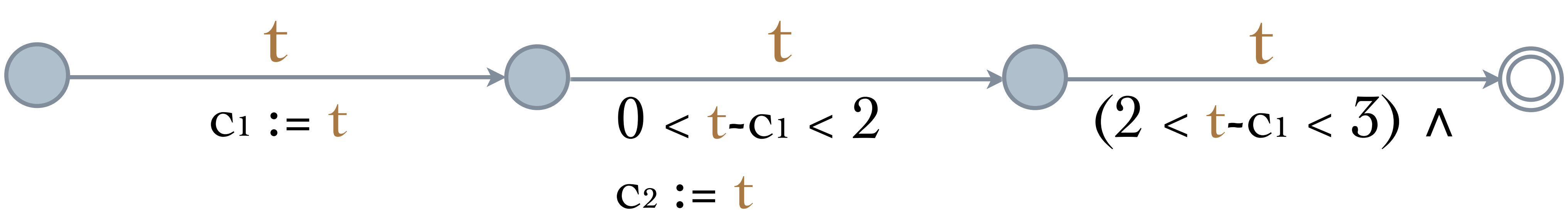




Timed automata are register automata

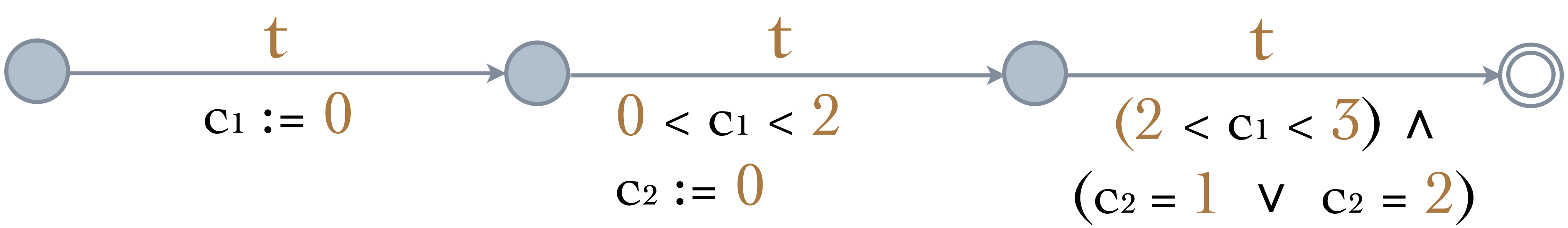
[Bojańczyk, L. 2012]

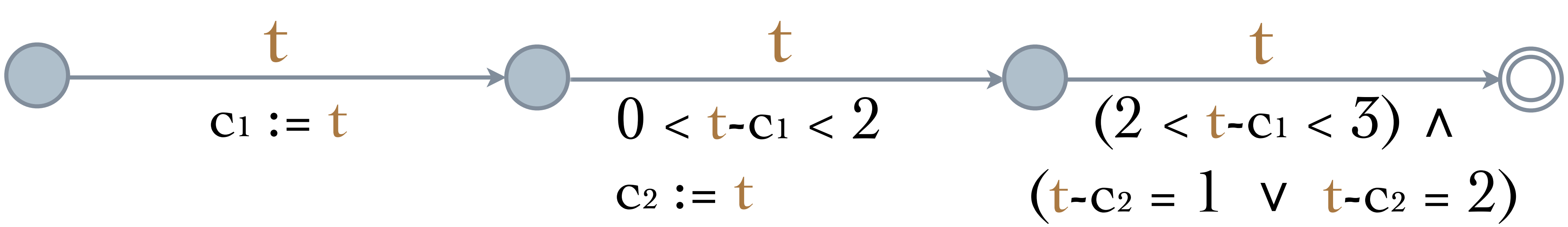




Timed automata are register automata

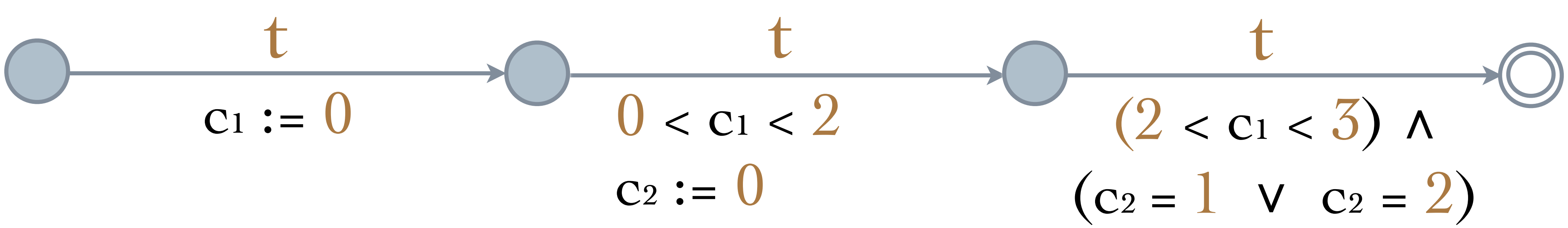
[Bojańczyk, L. 2012]

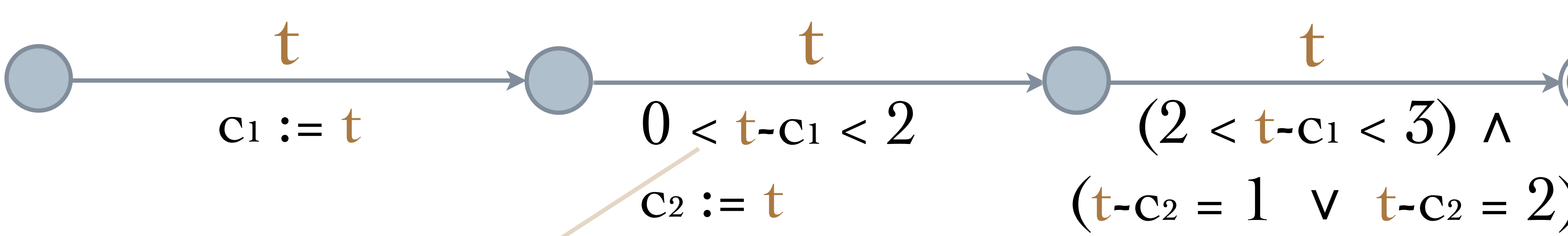




Timed automata are register automata

[Bojańczyk, L. 2012]



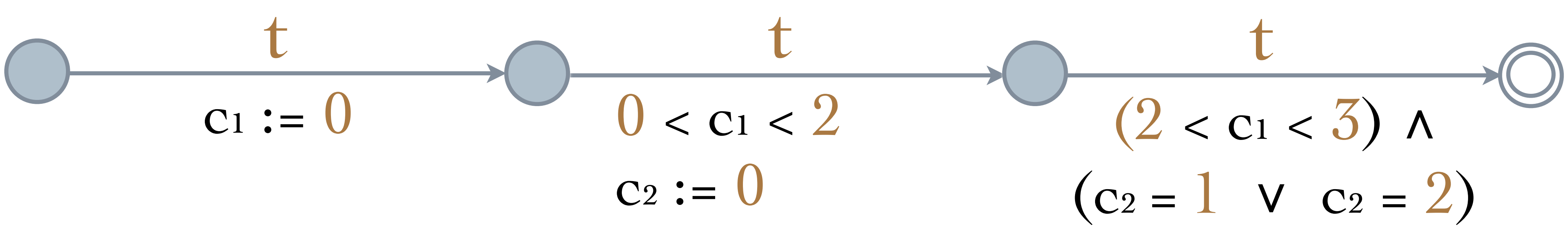


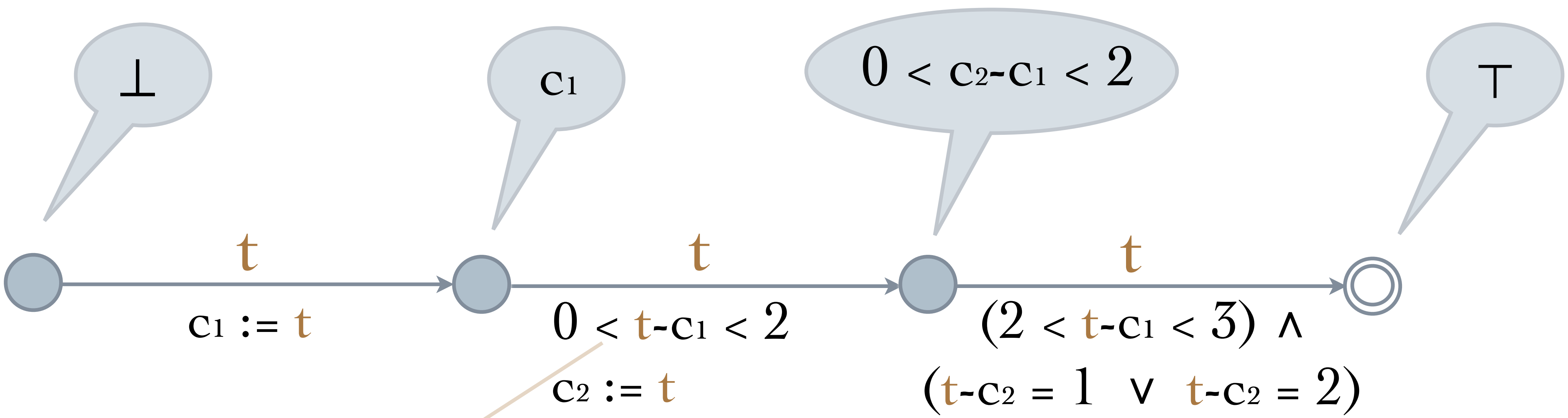
the guards use the structure $(\mathbb{R}, <, +1)$

e.g. $0 < t\text{-}c_1 < 2$ iff $c_1 < t < c_1+2$

Timed automata are register automata

[Bojańczyk, L. 2012]



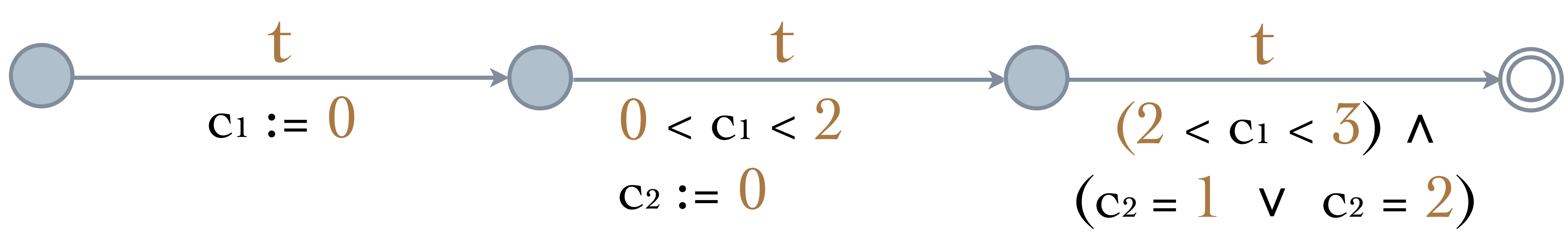


the guards use the structure $(R, <, +1)$

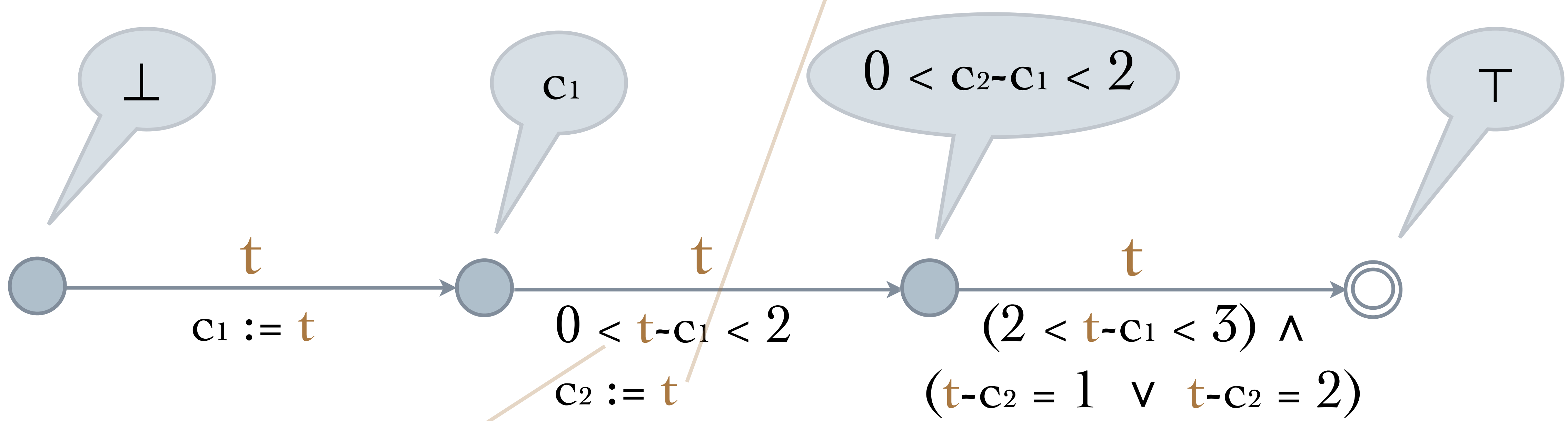
e.g. $0 < t - c_1 < 2$ iff $c_1 < t < c_1 + 2$

Timed automata are register automata

[Bojańczyk, L. 2012]



the only modifications of a clock: $c := t$



the guards use the structure $(\mathbb{R}, <, +1)$

e.g. $0 < t - c_1 < 2$ iff $c_1 < t < c_1 + 2$

FO(<, +1)-definable sets

dimension

FO(<, +1) formula $\phi(x_1, \ldots, x_n)$ defines a subset of n-tuples of reals, for instance

$$\phi(x_1, x_2) \quad \equiv \quad \exists x_3 \; (x_1 < x_3 \;\wedge\; x_2 = x_3 + 3)$$

FO-definable sets

dimension

FO(<, +1) formula $\phi(x_1, \ldots, x_n)$ defines a subset of n-tuples of reals, for instance

$$\phi(x_1, x_2) \quad \equiv \quad \exists x_3 \; (x_1 < x_3 \; \wedge \; x_2 = x_3 + 3)$$

FO-definable sets

dimension

FO(<, +1) formula $\phi(x_1, \ldots, x_n)$ defines a subset of n-tuples of reals, for instance

$$\phi(x_1, x_2) \quad \equiv \quad \exists x_3 \; (x_1 < x_3 \; \wedge \; x_2 = x_3 + 3)$$

FO(<, +1) = QF(<, +1) =

FO-definable sets

dimension

FO(<, +1) formula $\phi(x_1, \ldots, x_n)$ defines a subset of n-tuples of reals, for instance

$$\phi(x_1, x_2) \quad \equiv \quad \exists x_3 \; (x_1 < x_3 \;\wedge\; x_2 = x_3 + 3)$$

FO(<, +1) = QF(<, +1) = $\displaystyle\bigvee_{\text{finite}} \underbrace{\bigwedge_{\text{finite}} x_i - x_j \in I}_{\text{zone}}$

FO-definable sets

dimension

FO(<, +1) formula $\phi(x_1, \ldots, x_n)$ defines a subset of n-tuples of reals, for instance

$$\phi(x_1, x_2) \quad \equiv \quad \exists x_3 \; (x_1 < x_3 \; \wedge \; x_2 = x_3 + 3)$$

$$\text{FO}(<, +1) = \text{QF}(<, +1) = \bigvee_{\text{finite}} \underbrace{\bigwedge_{\text{finite}} x_i - x_j \in I}_{\text{zone}}$$

for instance

$$\phi(x_1, x_2) \quad \equiv \quad x_1 + 3 < x_2 \quad \equiv \quad x_2 - x_1 \in (3, \infty)$$

FO-definable NFA

- alphabet A
- states Q
- transitions $\delta \subseteq Q \times A \times Q$
- $I, F \subseteq Q$

FO-definable NFA

- alphabet A
- states Q
- transitions $\delta \subseteq Q \times A \times Q$
- $I, F \subseteq Q$

definable in FO(<, +1)

FO-definable NFA

- alphabet A $\phi_A(x_1, \ldots, x_n)$
- states Q $\phi_Q(x_1, \ldots, x_m)$
- transitions δ ⊆ Q × A × Q $\phi_\delta(x_1, \ldots, x_{m+n+m})$
- I, F ⊆ Q $\phi_I(x_1, \ldots, x_m),\ \phi_F(x_1, \ldots, x_m)$

FO-definable NFA

- alphabet A
- states Q
- transitions $\delta \subseteq Q \times A \times Q$
- I, F $\subseteq$ Q

$\Big\}$ definable in FO(<, +1)

$$\phi_A(x_1, \ldots, x_n)$$
$$\phi_Q(x_1, \ldots, x_m)$$
$$\phi_\delta(x_1, \ldots, x_{m+n+m})$$
$$\phi_I(x_1, \ldots, x_m), \ \phi_F(x_1, \ldots, x_m)$$

FO-definable NFA

- alphabet A
- states Q
- transitions $\delta \subseteq Q \times A \times Q$
- $I, F \subseteq Q$

$\Big\}$ definable in FO(<, +1)

$$\phi_A(x_1, \ldots, x_n)$$
$$\phi_Q(x_1, \ldots, x_m)$$
$$\phi_\delta(x_1, \ldots, x_{m+n+m})$$
$$\phi_I(x_1, \ldots, x_m), \ \phi_F(x_1, \ldots, x_m)$$

Runs, acceptance, language recognized, etc. are defined exactly as for classical NFA!

FO-definable NFA

- alphabet A
- states Q
- transitions $\delta \subseteq Q \times A \times Q$
- $I, F \subseteq Q$

$\left.\begin{array}{l}\text{alphabet } A \\ \text{states } Q\end{array}\right\}$ orbit-finite

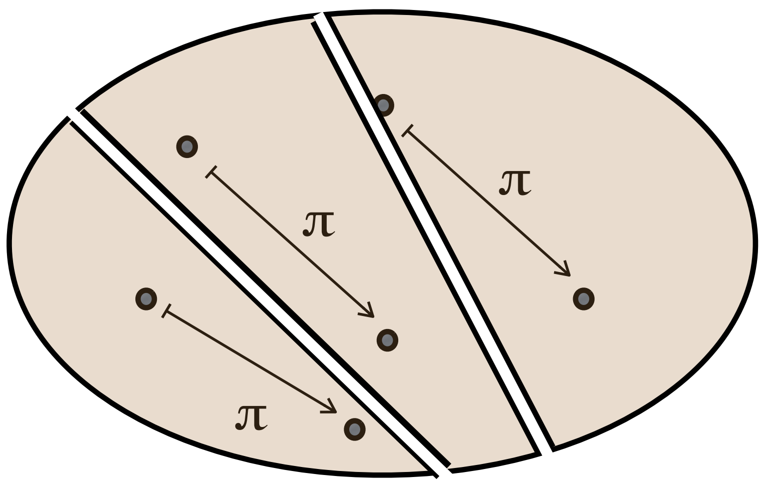
$\left.\begin{array}{l}\text{alphabet } A \\ \text{states } Q \\ \text{transitions } \delta \\ I, F\end{array}\right\}$ definable in FO(<, +1)

$$\phi_A(x_1, \ldots, x_n)$$
$$\phi_Q(x_1, \ldots, x_m)$$
$$\phi_\delta(x_1, \ldots, x_{m+n+m})$$
$$\phi_I(x_1, \ldots, x_m), \ \phi_F(x_1, \ldots, x_m)$$

Runs, acceptance, language recognized, etc. are defined exactly as for classical NFA!

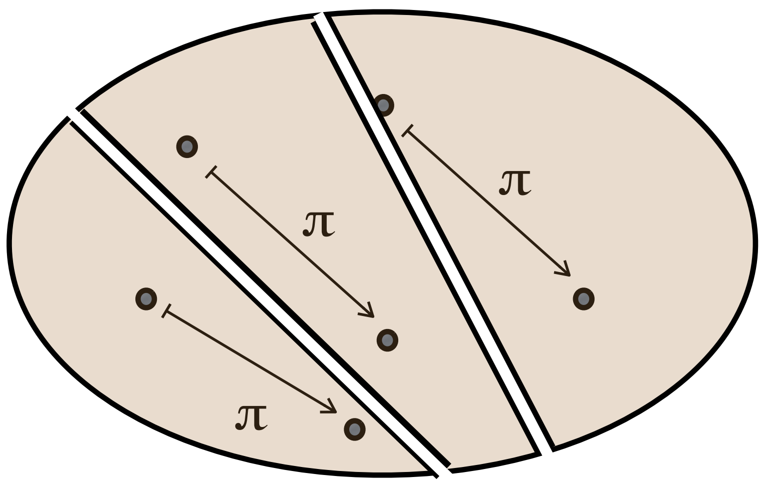
Orbit-finiteness

Automorphisms π of (R, <, +1) act on a definable set thus splitting it into orbits.



Orbit-finiteness

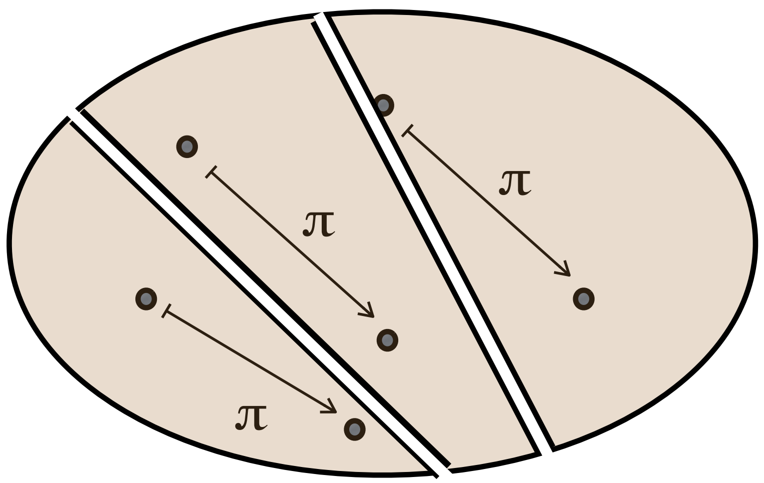
Automorphisms π of (R, <, +1) act on a definable set thus splitting it into orbits.



For instance, (-1, ⅓) and (3, 4⅓) and (1⅓, 3) are in the same orbit.

Orbit-finiteness

Automorphisms π of (R, <, +1) act on a definable set thus splitting it into orbits.



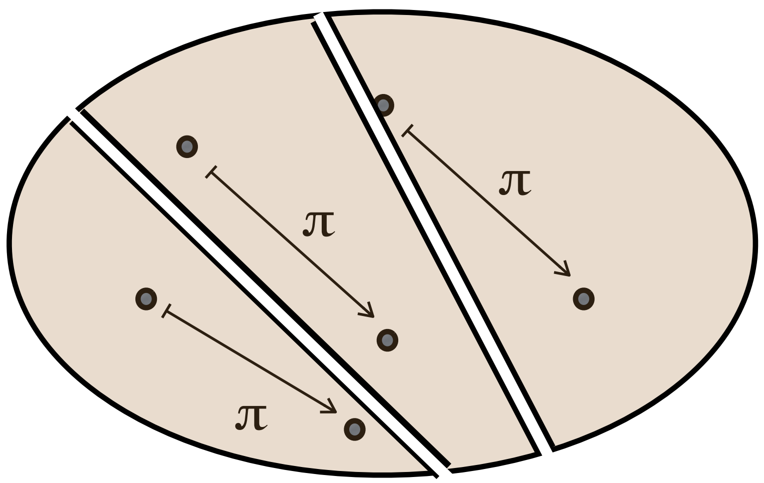
For instance, (-1, ⅓) and (3, 4⅓) and (1⅓, 3) are in the same orbit.

Example:

$x_1 + 3 < x_2 \quad \equiv \quad x_2 - x_1 \in (3, \infty)$ orbit-infinite

Orbit-finiteness

Automorphisms π of (R, <, +1) act on a definable set thus splitting it into orbits.



For instance, (-1, ⅓) and (3, 4⅓) and (1⅓, 3) are in the same orbit.

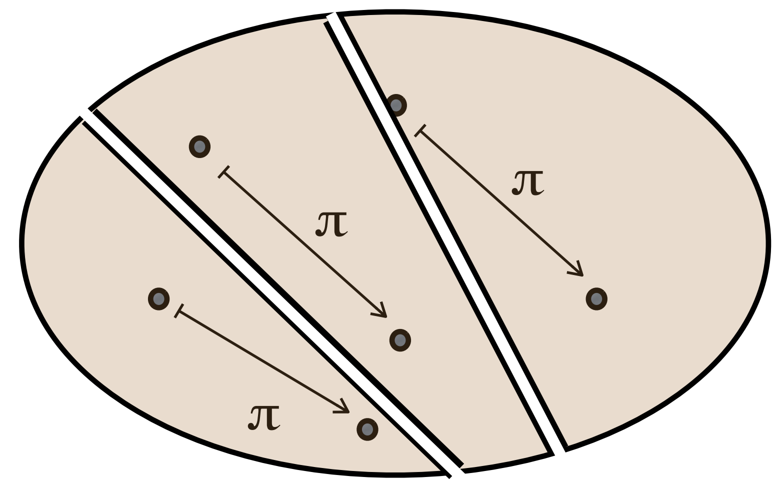
Example:

$x_1 + 3 < x_2 \quad \equiv \quad x_2 - x_1 \in (3, \infty)$ orbit-infinite

$x_1 + 3 < x_2 \leq x_1 + 7 \quad \equiv \quad x_2 - x_1 \in (3, 7]$ orbit-finite

Orbit-finiteness

Automorphisms π of (R, <, +1) act on a definable set thus splitting it into orbits.



For instance, (-1, ⅓) and (3, 4⅓) and (1⅓, 3) are in the same orbit.

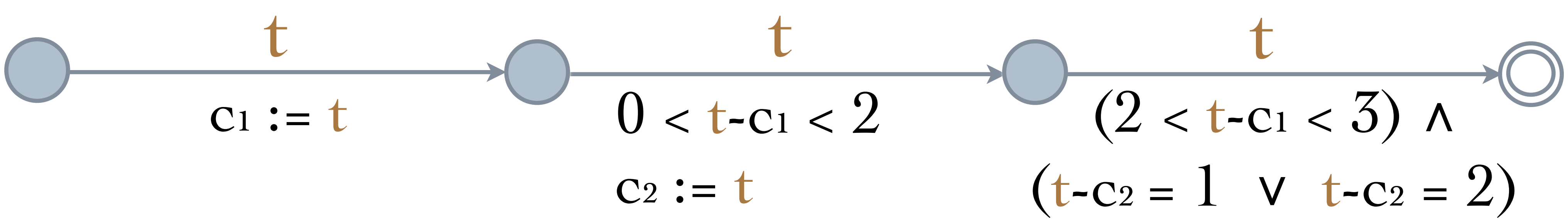
Example:

$$x_1 + 3 < x_2 \quad \equiv \quad x_2 - x_1 \in (3, \infty)$$ orbit-infinite

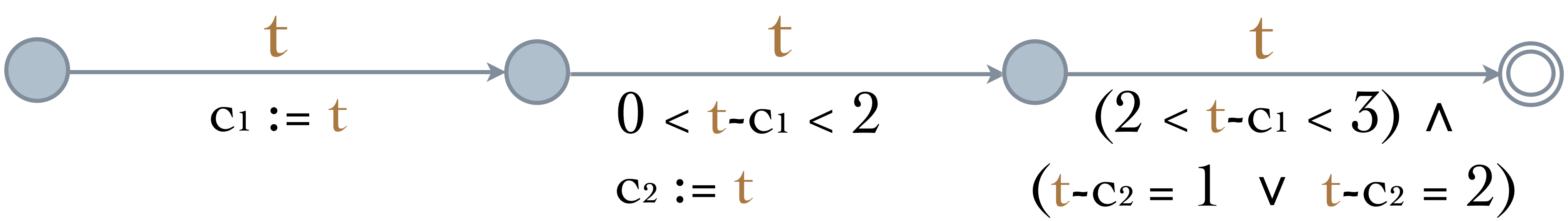
$$x_1 + 3 < x_2 \leq x_1 + 7 \quad \equiv \quad x_2 - x_1 \in (3, 7]$$ orbit-finite

An FO-definable set is orbit-finite
iff
it is defined using bounded intervals only

Register automata are FO-definable NFA

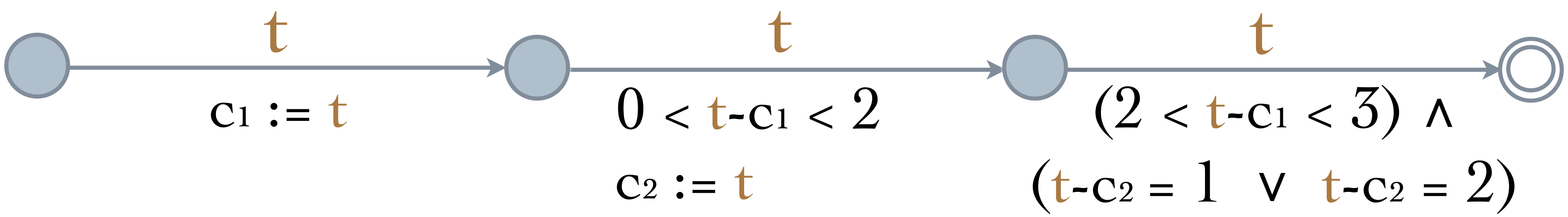


Register automata are FO-definable NFA



states: $Q = \{\bot\} \cup \{c_1 \in R\} \cup \{ (c_1, c_2) \in R \times R : 0 < c_2 - c_1 < 2 \} \cup \{\top\}$

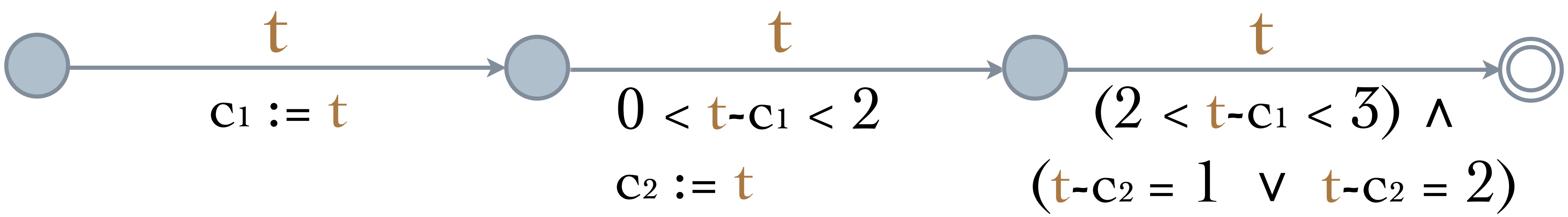
Register automata are FO-definable NFA



states: $Q = \{\bot\} \;\cup\; \{c_1 \in \mathbb{R}\} \;\cup\; \{\, (c_1, c_2) \in \mathbb{R} \times \mathbb{R} : 0 < c_2 - c_1 < 2 \,\} \;\cup\; \{\top\}$

$\phi_Q(c_0, c_1, c_2) \equiv c_0 = c_1 = c_2 \vee c_0 + 1 = c_1 = c_2 \vee c_0 + 2 = c_1 < c_2 < c_1 + 2 \vee c_0 + 3 = c_1 = c_2$

Register automata are FO-definable NFA

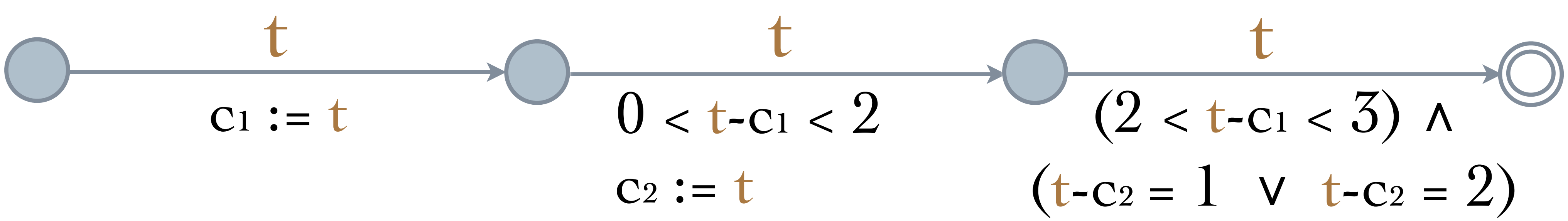


states: $Q = \{\bot\} \cup \{c_1 \in R\} \cup \{ (c_1, c_2) \in R \times R : 0 < c_2 - c_1 < 2 \} \cup \{\top\}$

$\phi_Q(c_0, c_1, c_2) \equiv c_0 = c_1 = c_2 \vee c_0 + 1 = c_1 = c_2 \vee c_0 + 2 = c_1 < c_2 < c_1 + 2 \vee c_0 + 3 = c_1 = c_2$

transitions: $\delta = \{ (\bot, t, c_1') : c_1' = t \} \cup$

Register automata are FO-definable NFA



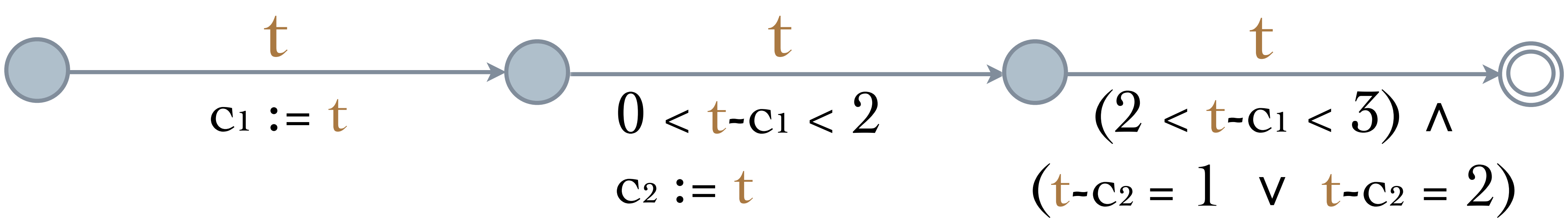
states: $Q = \{\perp\} \;\cup\; \{c_1 \in R\} \;\cup\; \{\, (c_1, c_2) \in R \times R : 0 < c_2 - c_1 < 2 \,\} \;\cup\; \{\top\}$

$\phi_Q(c_0, c_1, c_2) \equiv c_0 = c_1 = c_2 \lor c_0 + 1 = c_1 = c_2 \lor c_0 + 2 = c_1 < c_2 < c_1 + 2 \lor c_0 + 3 = c_1 = c_2$

transitions: $\delta = \{\, (\perp, t, c_1') : c_1' = t \,\} \;\cup$

$\{\, (c_1, t, (c_1', c_2')) : 0 < t - c_1 < 2 \land c_1 = c_1' \land c_2' = t \,\} \;\cup$

Register automata are FO-definable NFA



states: $Q = \{\bot\} \ \cup \ \{c_1 \in R\} \ \cup \ \{ (c_1, c_2) \in R \times R : 0 < c_2 - c_1 < 2 \} \ \cup \ \{\top\}$

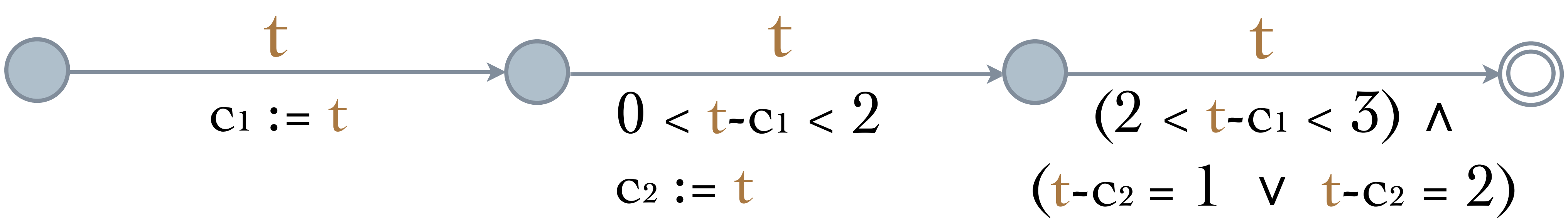
$\phi_Q(c_0, c_1, c_2) \equiv c_0 = c_1 = c_2 \vee c_0 + 1 = c_1 = c_2 \vee c_0 + 2 = c_1 < c_2 < c_1 + 2 \vee c_0 + 3 = c_1 = c_2$

transitions: $\delta = \{ (\bot, t, c_1') : c_1' = t \} \ \cup$

$\{ (c_1, t, (c_1', c_2')) : 0 < t - c_1 < 2 \wedge c_1 = c_1' \wedge c_2' = t \} \ \cup$

$\{ ((c_1, c_2), t, \top) : (2 < t - c_1 < 3) \wedge (t - c_2 = 1 \ \vee \ t - c_2 = 2) \}$

Register automata are FO-definable NFA



states: $Q = \{\bot\} \;\cup\; \{c_1 \in R\} \;\cup\; \{ (c_1, c_2) \in R \times R : 0 < c_2\text{-}c_1 < 2 \} \;\cup\; \{\top\}$

$\phi_Q(c_0, c_1, c_2) \equiv c_0 = c_1 = c_2 \vee c_0{+}1 = c_1 = c_2 \vee c_0{+}2 = c_1 < c_2 < c_1{+}2 \vee c_0{+}3 = c_1 = c_2$

transitions: $\delta = \{ (\bot, t, c_1') : c_1' = t \} \;\cup$

$\{ (c_1, t, (c_1', c_2')) : 0 < t\text{-}c_1 < 2 \wedge c_1 = c_1' \wedge c_2' = t \} \;\cup$

$\{ ((c_1, c_2), t, \top) : (2 < t\text{-}c_1 < 3) \wedge (t\text{-}c_2 = 1 \;\vee\; t\text{-}c_2 = 2) \}$

$\phi_\delta(c_0, c_1, c_2, t, c_0', c_1', c_2') \equiv \ldots$

Timed automata vs. FO-definable NFA

FO-definable NFA are like updatable timed automata [Bouyer, Duford, Fleury 2000], but:

Timed automata vs. FO-definable NFA

FO-definable NFA are like updatable timed automata [Bouyer, Duford, Fleury 2000], but:

- in every location, clock valuations are restricted by an orbit-finite constraint (invariant)

Timed automata vs. FO-definable NFA

FO-definable NFA are like updatable timed automata [Bouyer, Duford, Fleury 2000], but:

- in every location, clock valuations are restricted by an orbit-finite constraint (invariant)
- number of clocks may vary from one location to another

Timed automata vs. FO-definable NFA

FO-definable NFA are like updatable timed automata [Bouyer, Duford, Fleury 2000], but:

- in every location, clock valuations are restricted by an orbit-finite constraint (invariant)
- number of clocks may vary from one location to another
- the input needs not be monotonic (but can be enforced to be)

Timed automata vs. FO-definable NFA

FO-definable NFA are like updatable timed automata [Bouyer, Duford, Fleury 2000], but:

- in every location, clock valuations are restricted by an orbit-finite constraint (invariant)
- number of clocks may vary from one location to another
- the input needs not be monotonic (but can be enforced to be)
- alphabet letters may be a tuples of timestamps

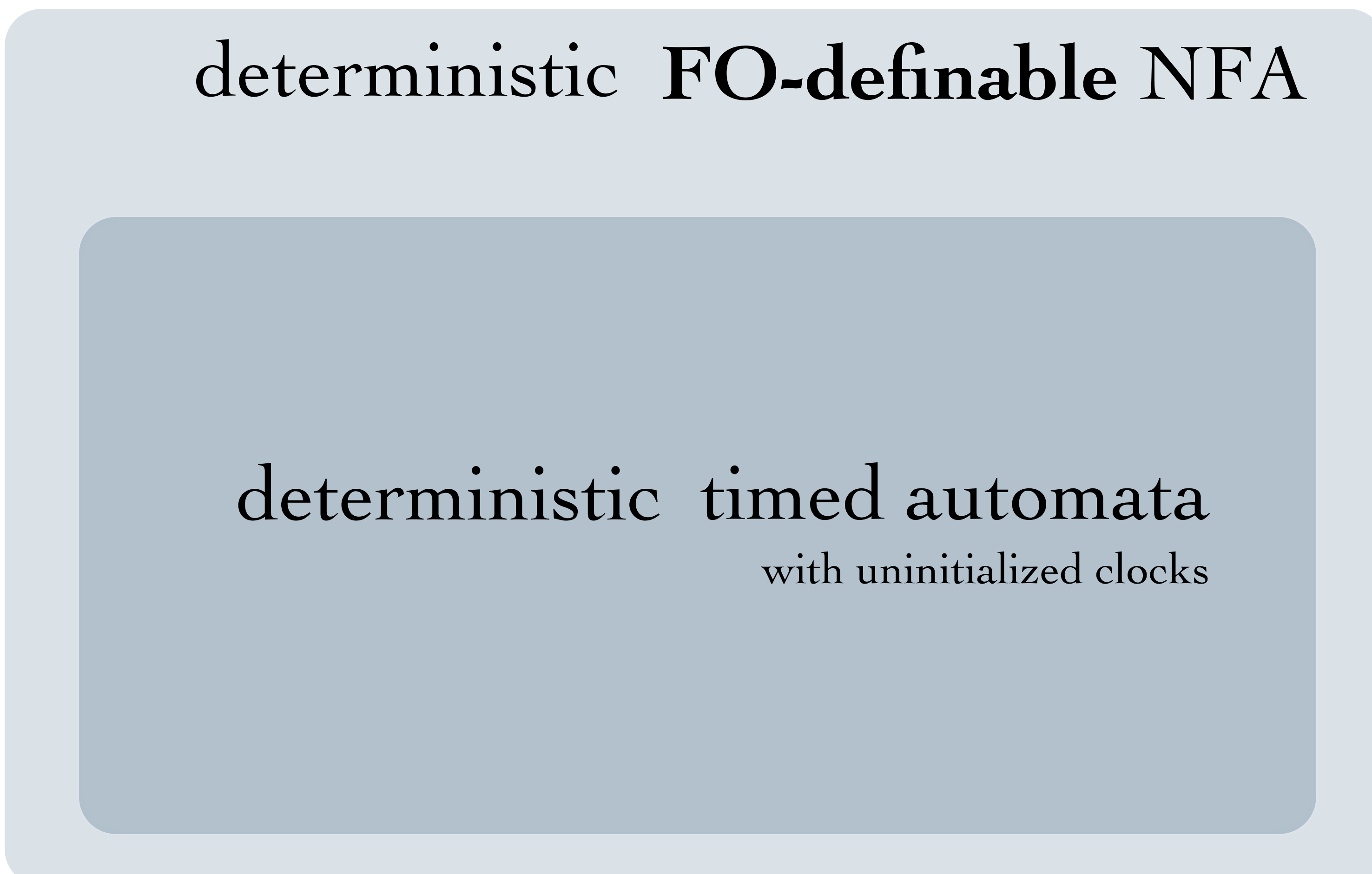
Timed automata vs. FO-definable NFA

FO-definable NFA

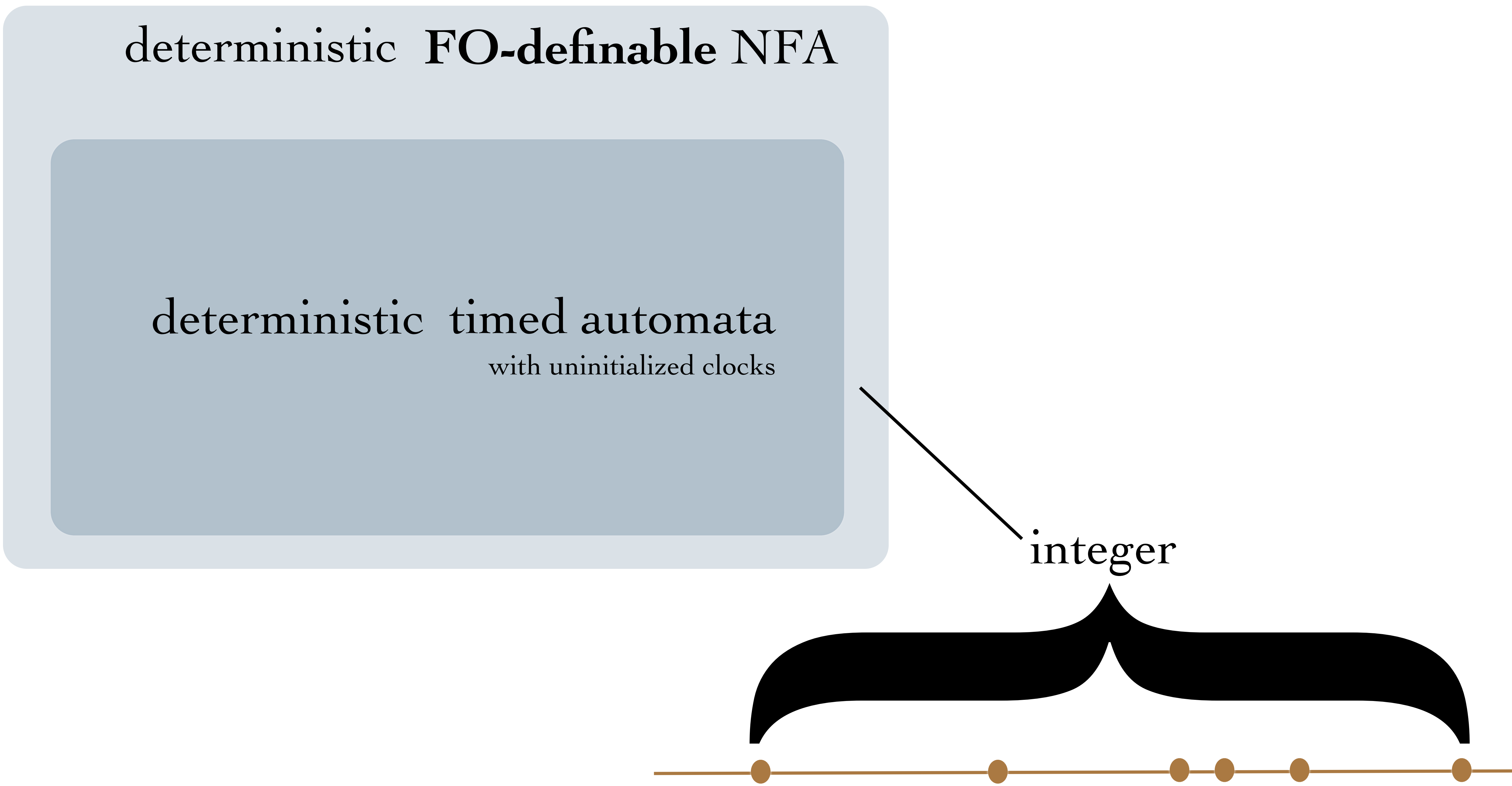
timed automata

with uninitialized clocks

Timed automata vs. FO-definable NFA



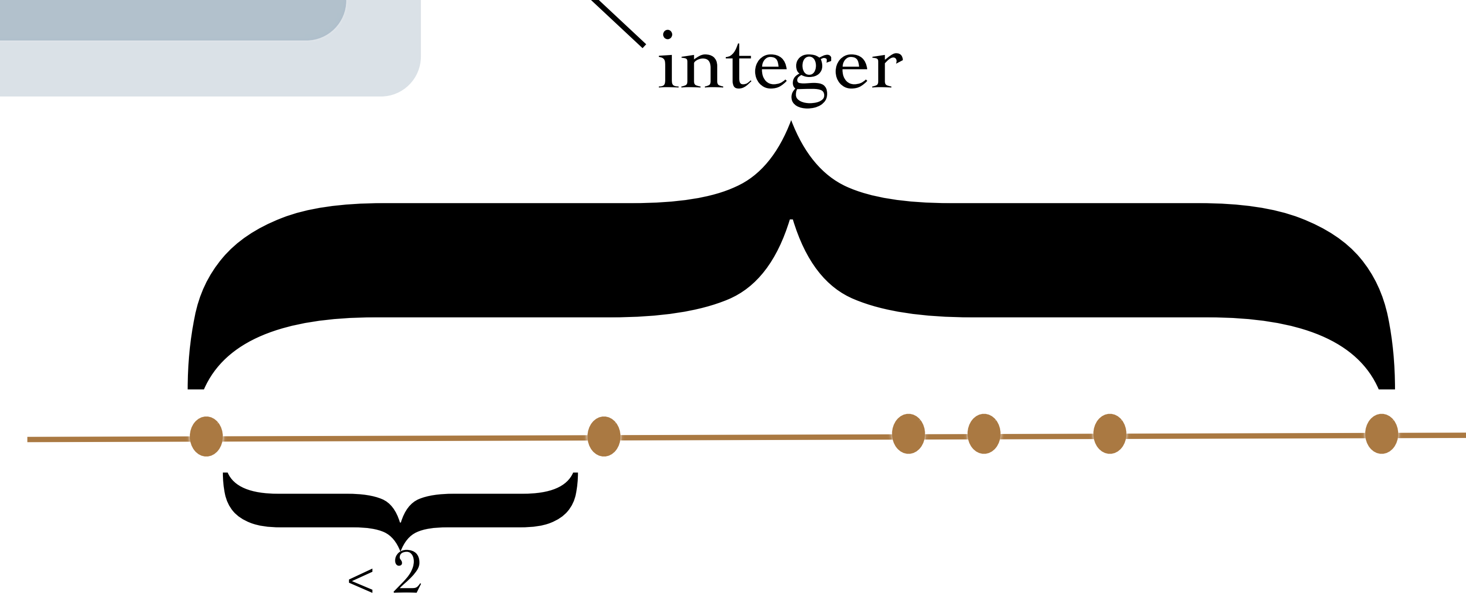
Timed automata vs. FO-definable NFA



Timed automata vs. FO-definable NFA

deterministic **FO-definable** NFA

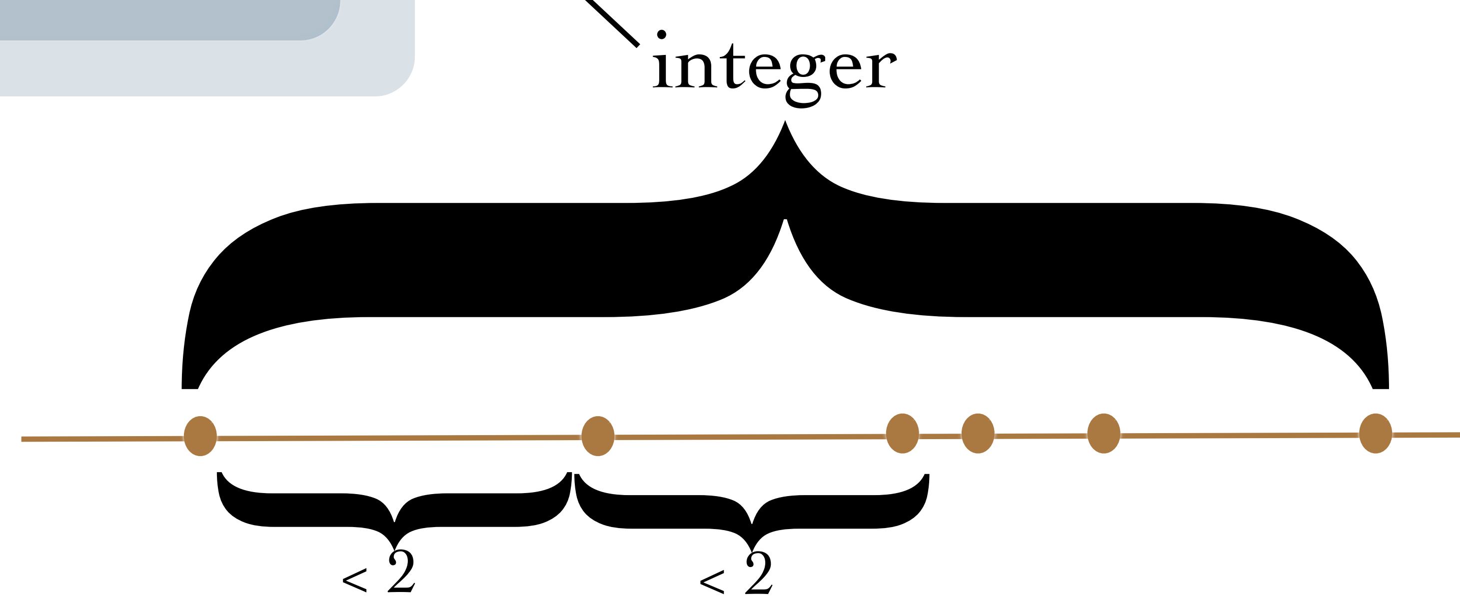
deterministic timed automata
with uninitialized clocks



Timed automata vs. FO-definable NFA

deterministic **FO-definable** NFA

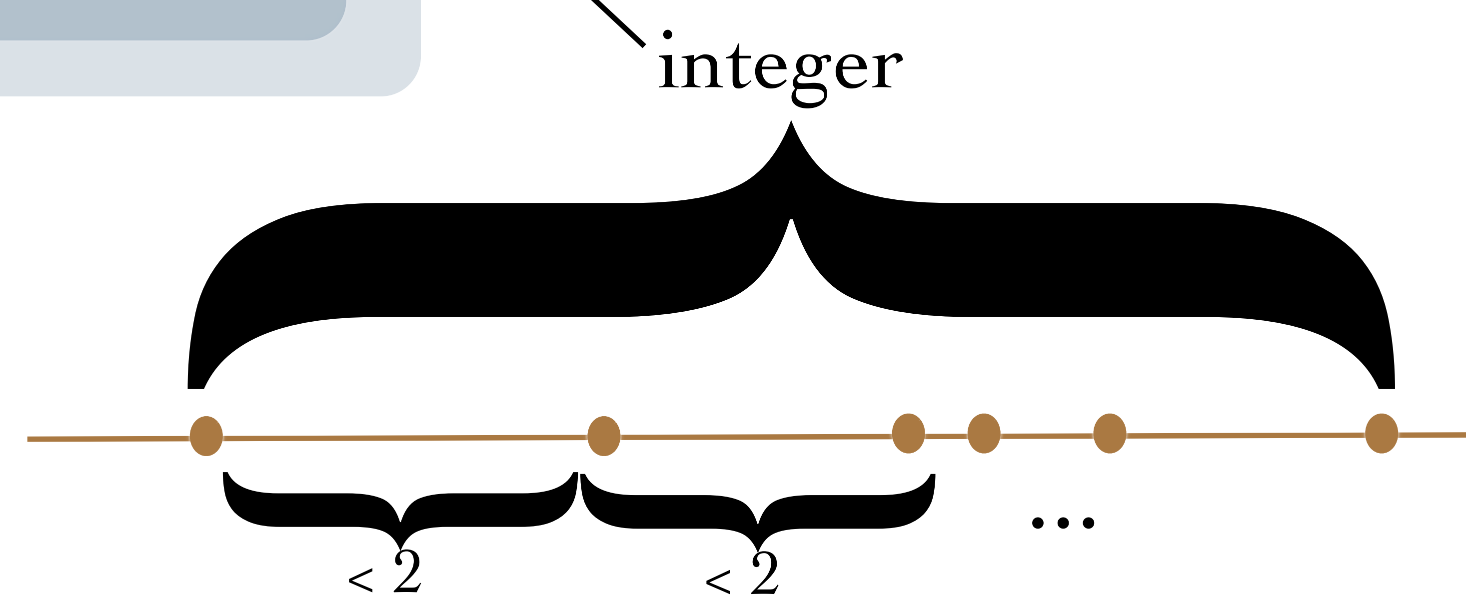
deterministic timed automata
with uninitialized clocks



Timed automata vs. FO-definable NFA

deterministic **FO-definable** NFA

deterministic timed automata
with uninitialized clocks

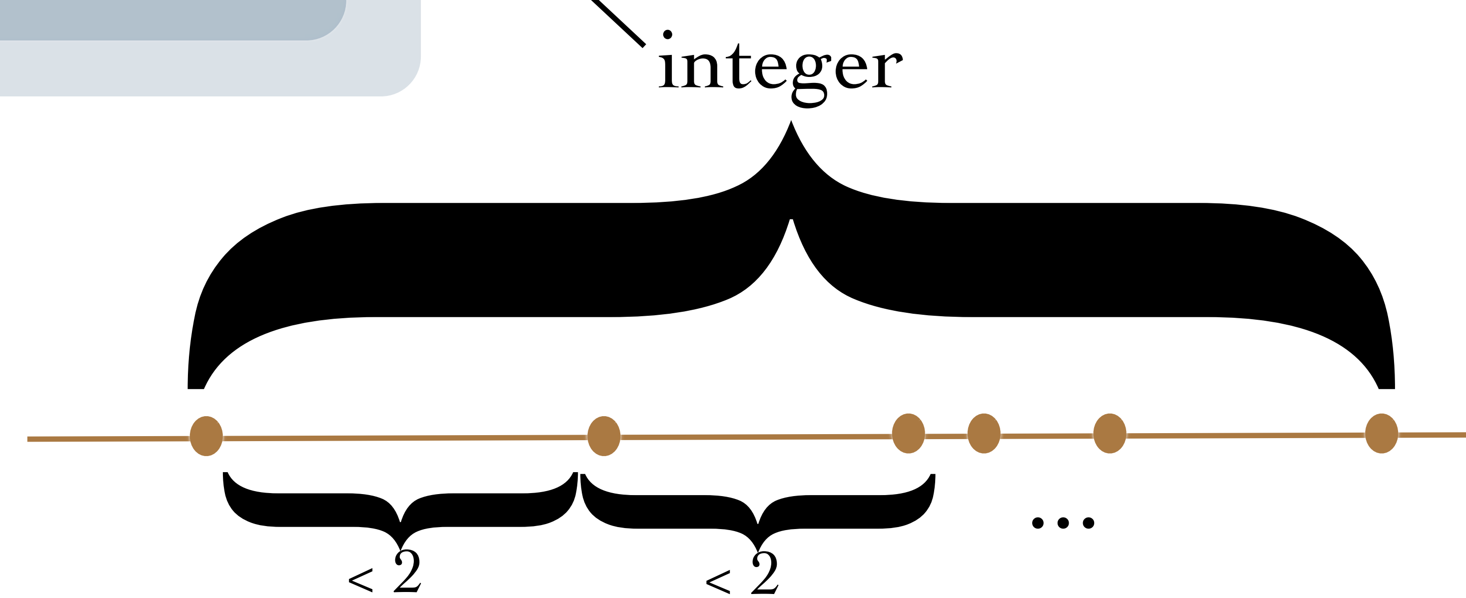


Timed automata vs. FO-definable NFA

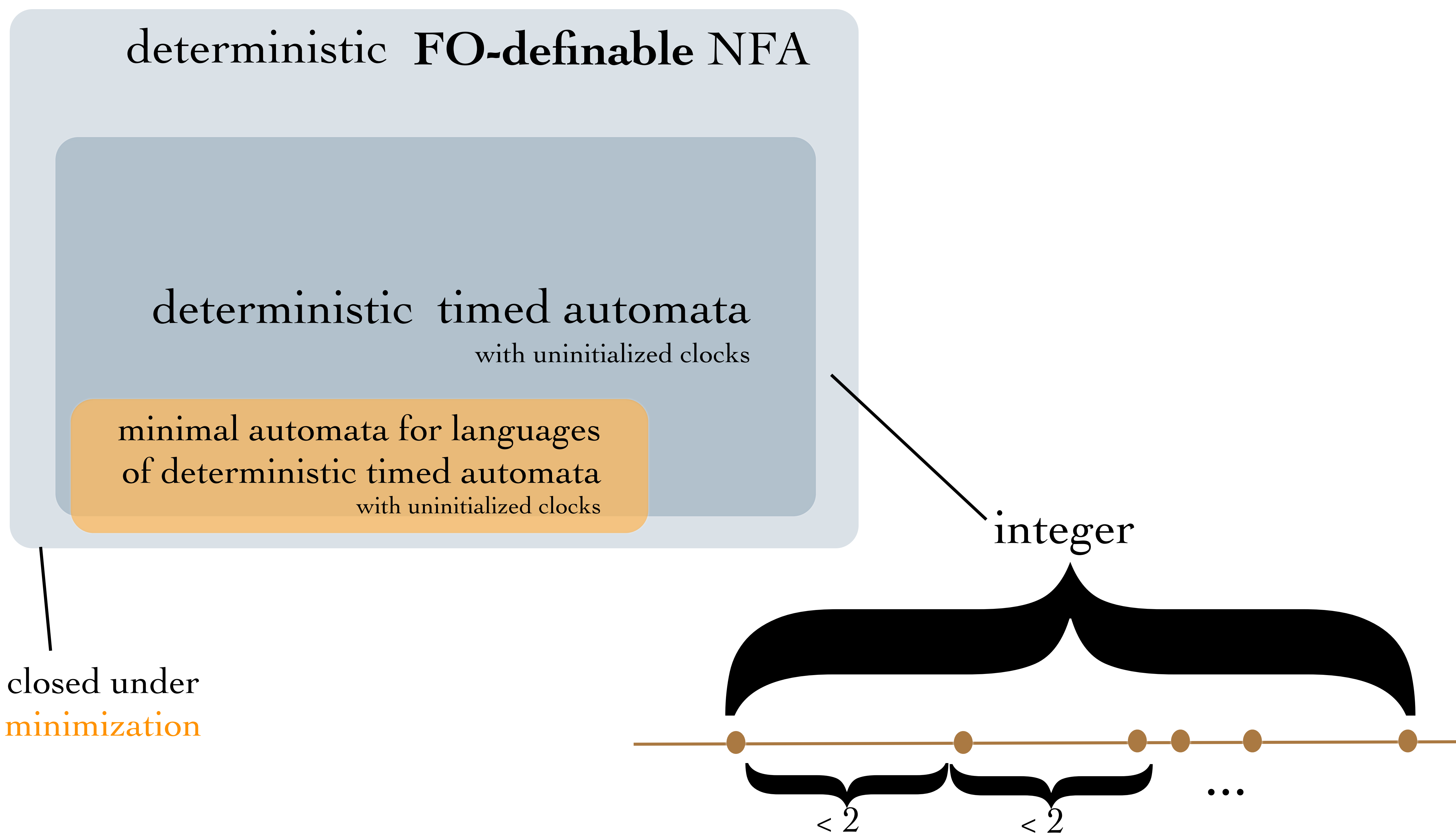
deterministic **FO-definable** NFA

deterministic timed automata
with uninitialized clocks

closed under
minimization

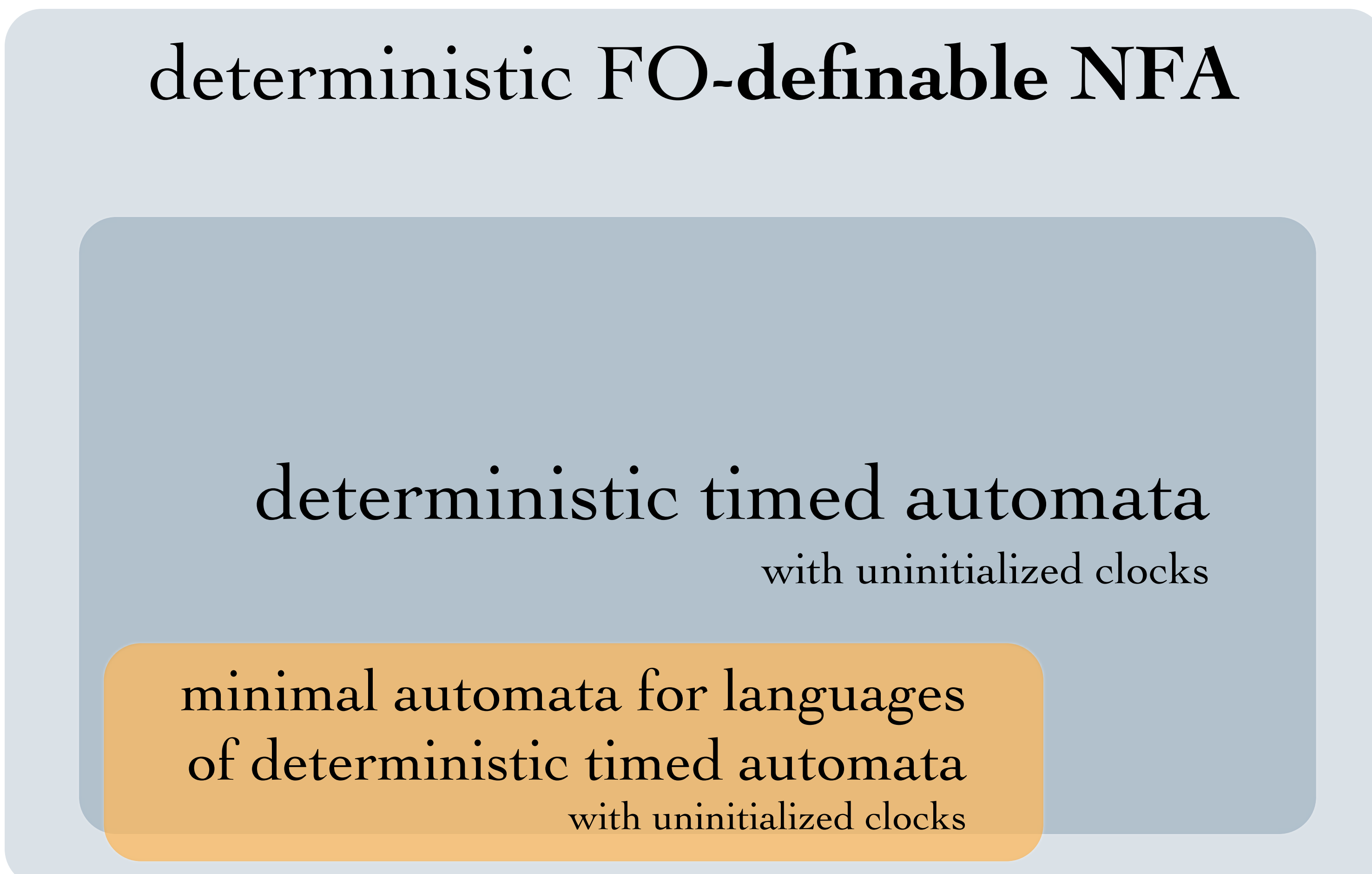


Timed automata vs. FO-definable NFA



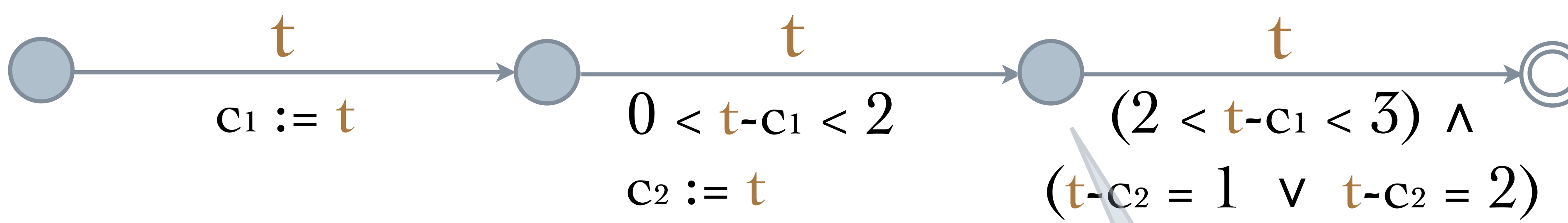
FO-definable DFA do minimize

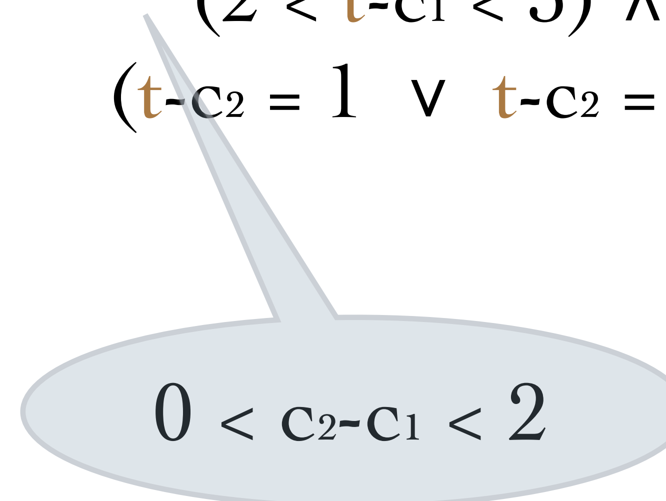
[Bojańczyk, L. 2012]

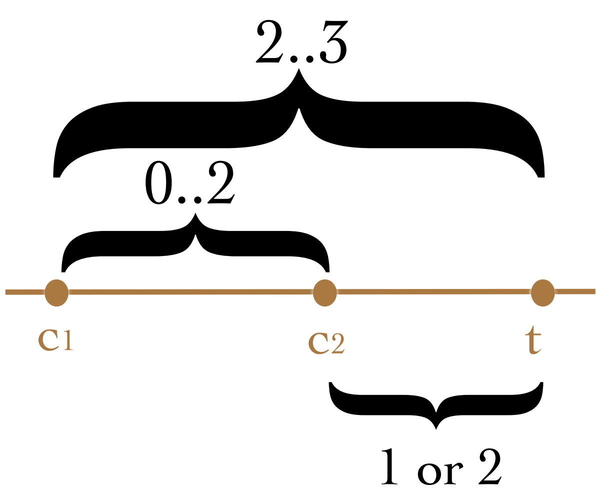


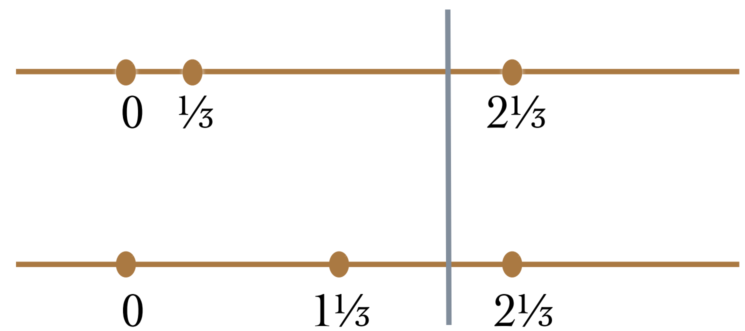
FO-definable DFA do minimize

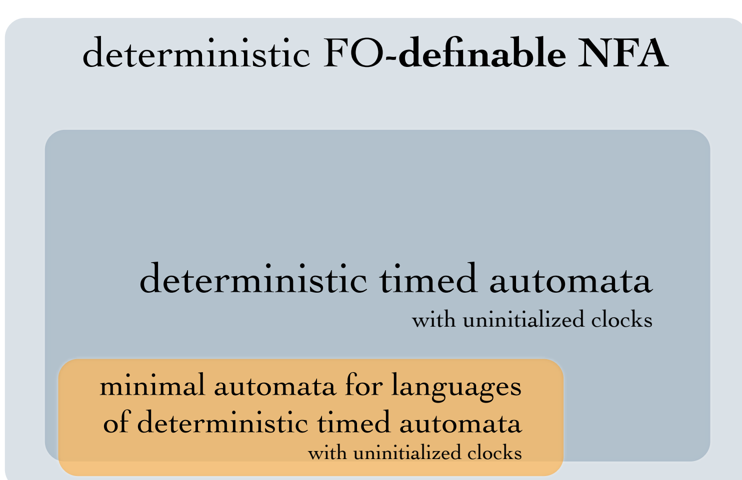
[Bojańczyk, L. 2012]





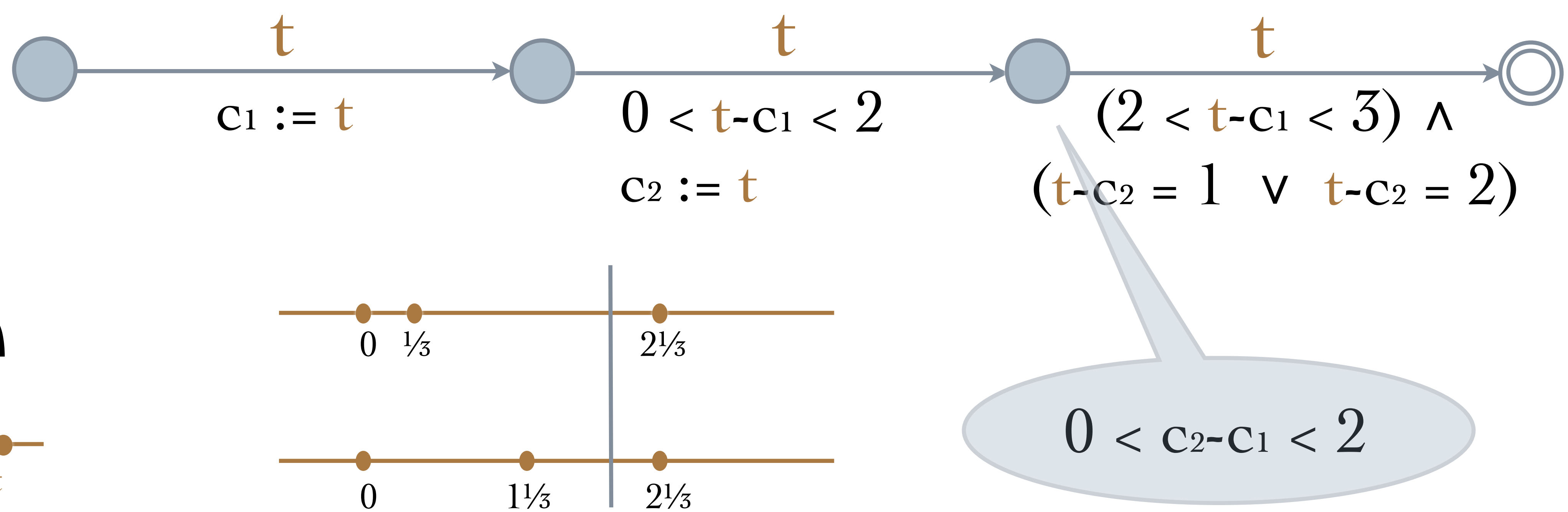


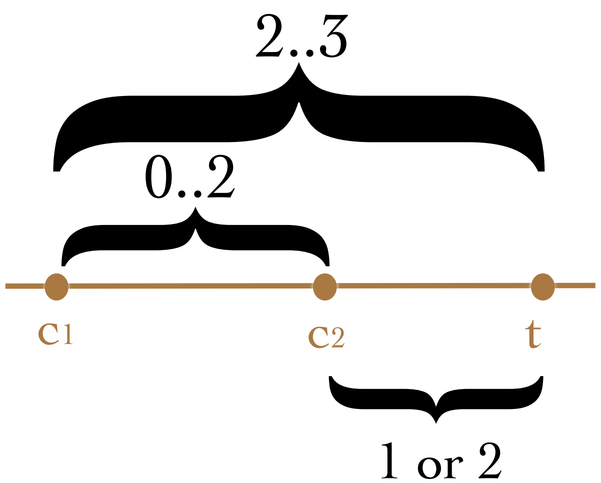


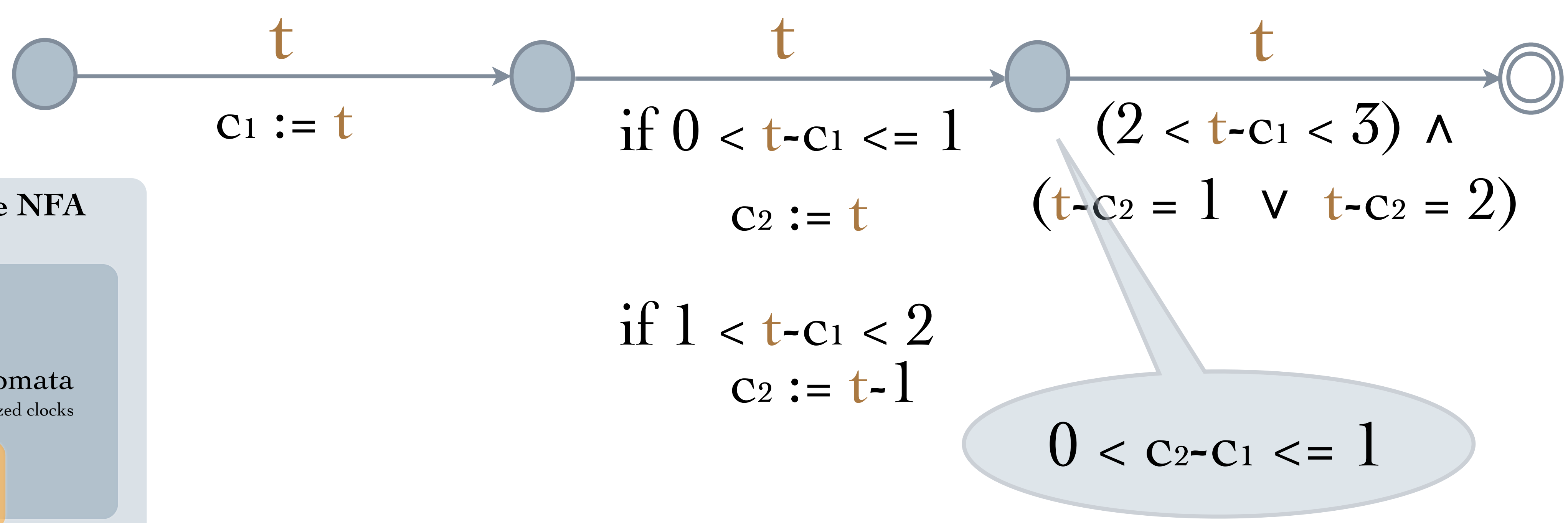


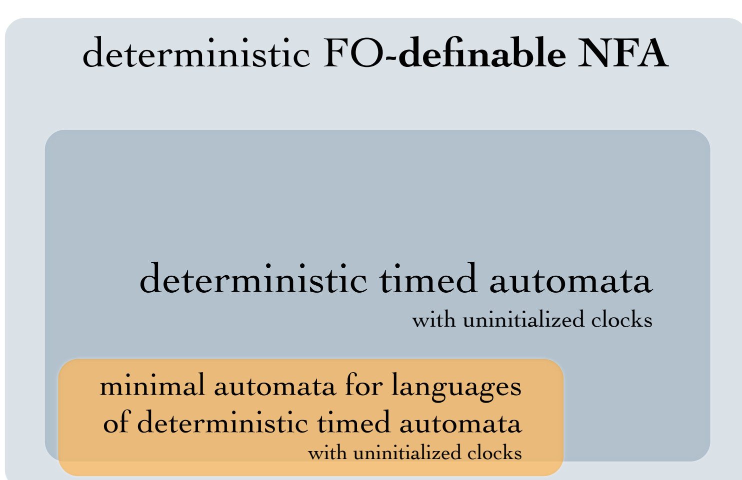
FO-definable DFA do minimize

[Bojańczyk, L. 2012]









Presburger NFA

[Bojańczyk, L. 2012]

Minimization holds also if FO(<, +1) is replaced by FO(<, +)

In search of lost definition

- Motivation
- FO-definable NFA
- FO-definable PDA
- The core problem: equations over sets of integers

In search of lost definition

- Motivation
- FO-definable NFA
- FO-definable PDA

PDA re-interpreted in FO-definable sets

- The core problem: equations over sets of integers

FO-definable PDA

- alphabet A
- states Q
- stack alphabet S
- push $\subseteq$ Q $\times$ A $\times$ Q $\times$ S
- pop $\subseteq$ Q $\times$ S $\times$ A $\times$ Q
- I, F $\subseteq$ Q

orbit-finite

definable in FO(<, +1)

FO-definable PDA

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- push $\subseteq$ Q × A × Q × S
- pop $\subseteq$ Q × S × A × Q
- I, F $\subseteq$ Q

$$\phi_A(x_1, \ldots, x_n)$$

$$\phi_Q(x_1, \ldots, x_m)$$

$$\phi_S(x_1, \ldots, x_k)$$

$$\phi_{\text{push}}(x_1, \ldots, x_{m+n+m+k})$$

$$\phi_{\text{pop}}(x_1, \ldots, x_{m+k+n+m})$$

$$\phi_I(x_1, \ldots, x_m), \; \phi_F(x_1, \ldots, x_m)$$

FO-definable PDA

- alphabet A
- states Q
- stack alphabet S
- push $\subseteq$ Q × A × Q × S
- pop $\subseteq$ Q × S × A × Q
- I, F $\subseteq$ Q

$\left.\begin{matrix}\text{alphabet A}\\ \text{states Q}\\ \text{stack alphabet S}\end{matrix}\right\}$ orbit-finite

$\left.\begin{matrix}\text{all of the above}\end{matrix}\right\}$ definable in FO(<, +1)

$\phi_A(x_1, \ldots, x_n)$
$\phi_Q(x_1, \ldots, x_m)$
$\phi_S(x_1, \ldots, x_k)$
$\phi_{\text{push}}(x_1, \ldots, x_{m+n+m+k})$
$\phi_{\text{pop}}(x_1, \ldots, x_{m+k+n+m})$
$\phi_I(x_1, \ldots, x_m),\ \phi_F(x_1, \ldots, x_m)$

Acceptance defined as for classical PDA.

Example

input alphabet: A = reals ⊎ {ε}

language: "ordered palindromes of even length over reals"

states:

stack alphabet:

transitions:

initial state:

accepting state:

Example

input alphabet:	$A = \text{reals} \uplus \{\varepsilon\}$
language:	"ordered palindromes of even length over reals"
states:	$Q = \text{reals} \uplus \{\text{init}, \text{finish}, \text{acc}\}$
stack alphabet:	
transitions:	
initial state:	init
accepting state:	acc

Example

input alphabet: A = reals ⊎ {ε}

language: "ordered palindromes of even length over reals"

states: Q = reals ⊎ {init, finish, acc}

stack alphabet: S = reals ⊎ {⊥}

transitions:

initial state: init

accepting state: acc

Example

input alphabet: A = reals ⊎ {ε}

language: "ordered palindromes of even length over reals"

states: Q = reals ⊎ {init, finish, acc}

stack alphabet: S = reals ⊎ {⊥}

transitions: push $\subseteq Q \times A \times Q \times S$

(init, ε, t, ⊥)	
(t, u, u, u)	t < u
(t, u, finish, u)	t < u

in state init, without reading input, change state to an arbitrary real t, and push ⊥ on stack

pop $\subseteq Q \times S \times A \times Q$

(finish, t, t, finish)	
(finish, ⊥, ε, acc)	

initial state: init

accepting state: acc

Example

input alphabet: $A = \text{reals} \uplus \{\varepsilon\}$

language: "ordered palindromes of even length over reals"

states: $Q = \text{reals} \uplus \{\text{init}, \text{finish}, \text{acc}\}$

stack alphabet: $S = \text{reals} \uplus \{\perp\}$

transitions: push $\subseteq Q \times A \times Q \times S$

(init, ε, t, ⊥)	
(t, u, u, u)	$t < u$
(t, u, finish, u)	$t < u$

in state finish, pop a real t from stack, read the same t from input, and stay in the same state

pop $\subseteq Q \times S \times A \times Q$

(finish, t, t, finish)	
(finish, ⊥, ε, acc)	

initial state: init

accepting state: acc

FO-definable prefix rewriting

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- $\rho \subseteq Q \times S^* \times A \times Q \times S^*$
- $I, F \subseteq Q$

definable in FO(<, +1)

FO-definable prefix rewriting

- alphabet A
- states Q
- stack alphabet S
- $\rho \subseteq Q \times S^{\leq n} \times A \times Q \times S^{\leq m}$
- $I, F \subseteq Q$

orbit-finite

definable in FO(<, +1)

FO-definable prefix rewriting

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- $\rho \subseteq Q \times S^{\leq n} \times A \times Q \times S^{\leq m}$
- $I, F \subseteq Q$

definable in FO(<, +1)

Acceptance defined as for classical prefix rewriting.

FO-definable context-free grammars

- nonterminal symbols S
- terminal symbols A

$\}$ orbit-finite

- an initial nonterminal symbol
- $\rho \subseteq S \times (S \uplus A)^*$

$\}$ definable in FO(<, +1)

FO-definable context-free grammars

- nonterminal symbols S
- terminal symbols A

orbit-finite

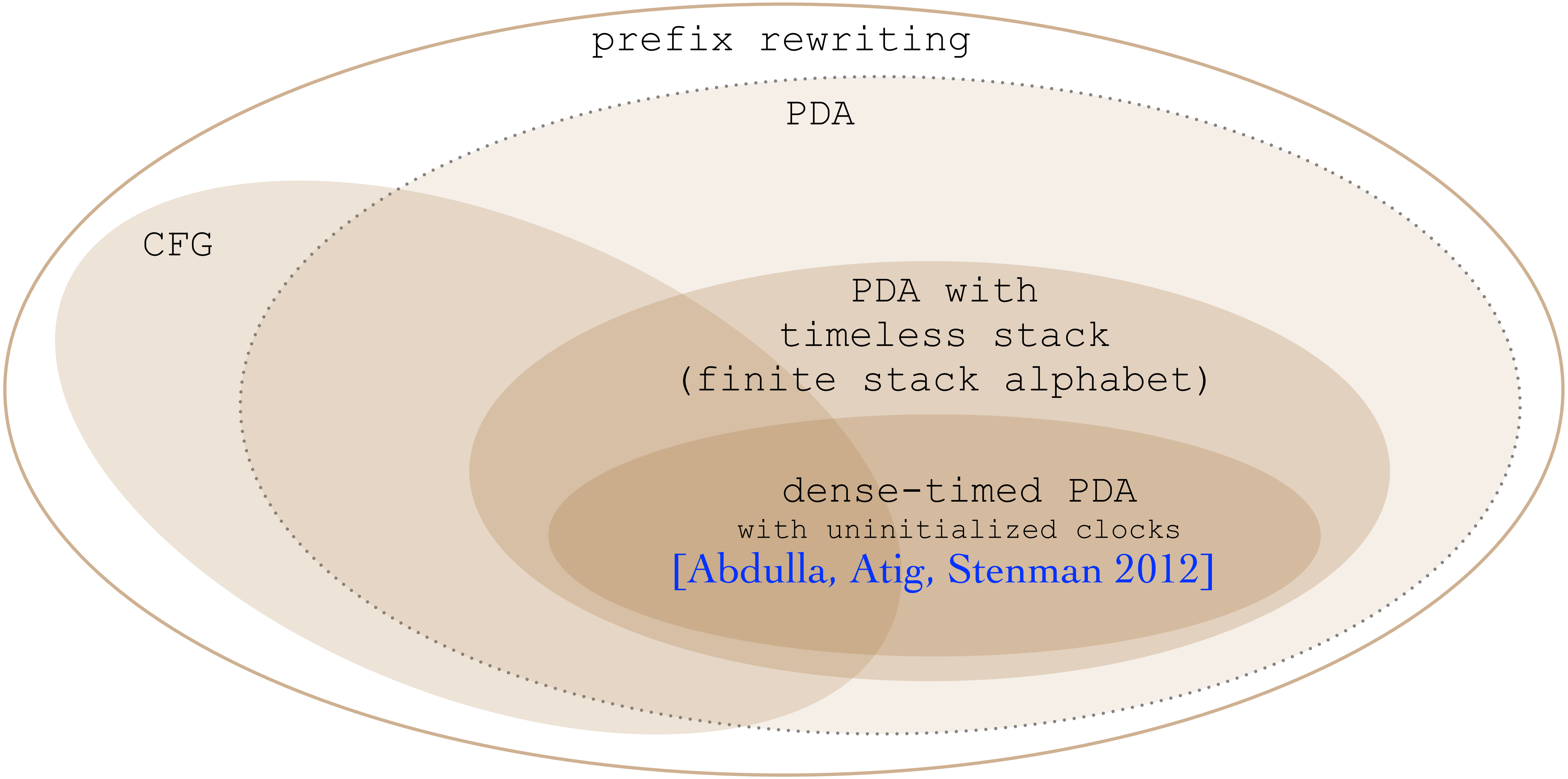
- an initial nonterminal symbol
- $\rho \subseteq S \times (S \uplus A)^{\leq n}$

definable in FO(<, +1)

Generated language defined as for classical PDA.

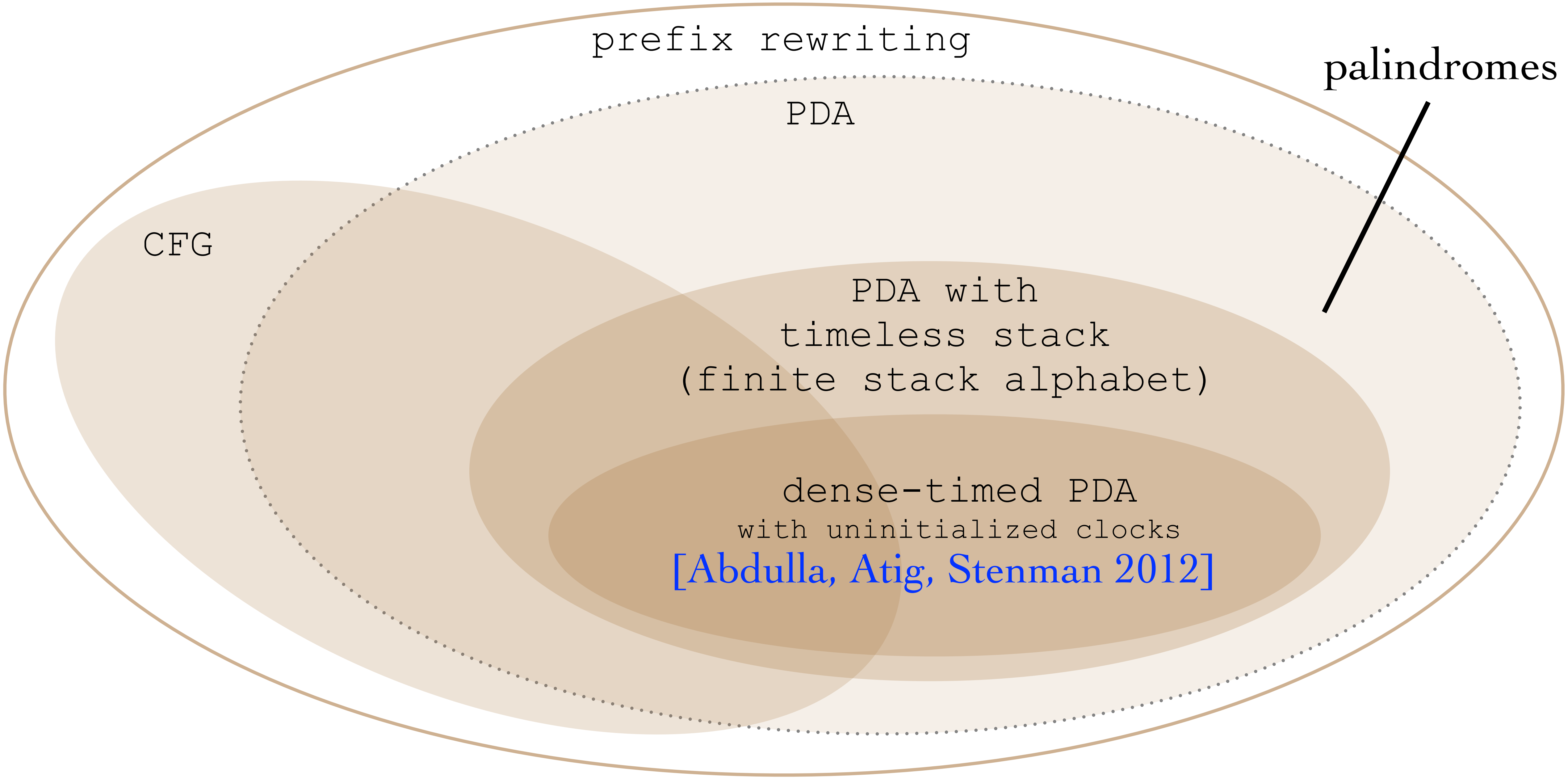
Expressiveness of FO-definable models

[Clemente, L. 2015]



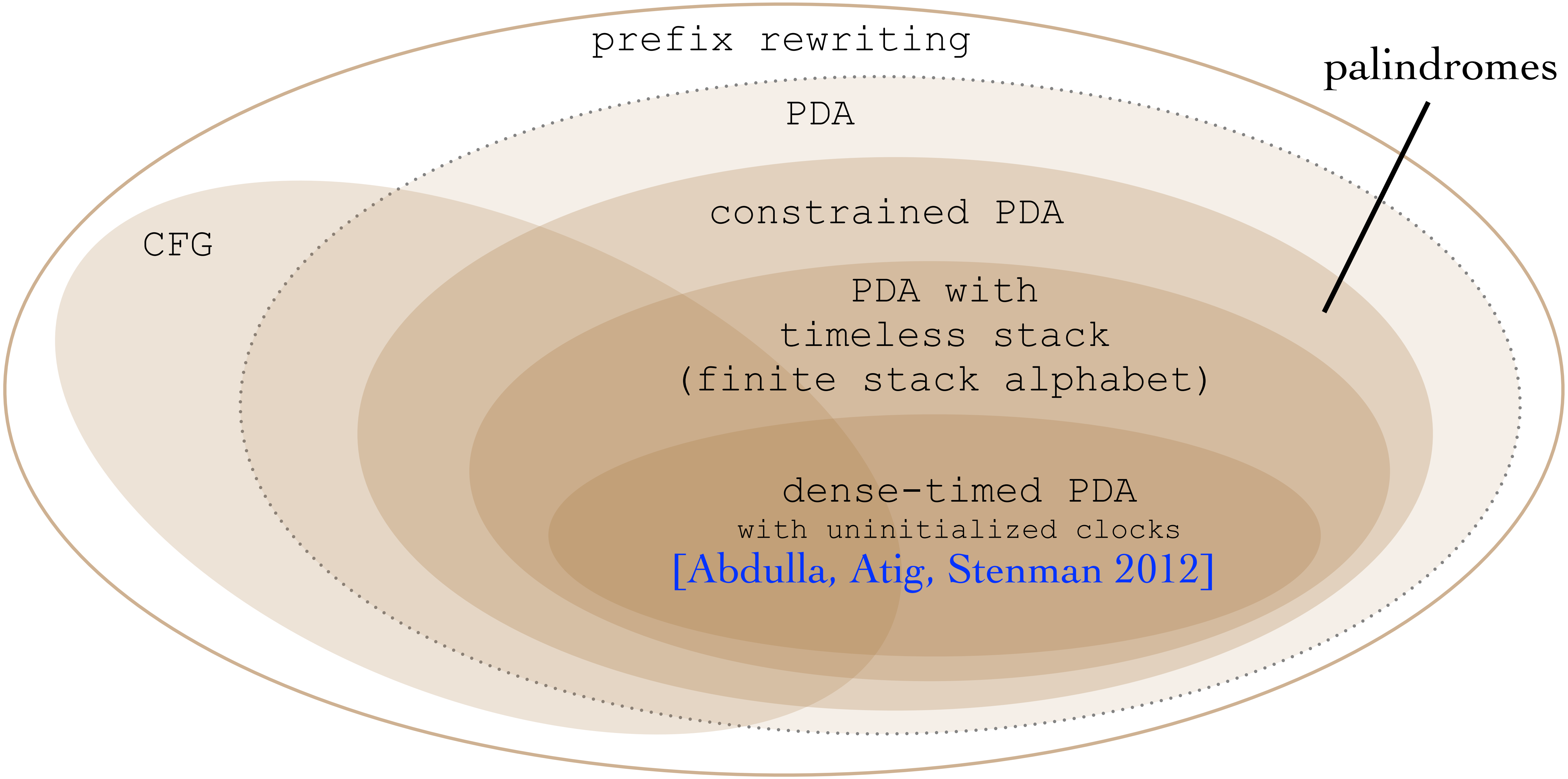
Expressiveness of FO-definable models

[Clemente, L. 2015]



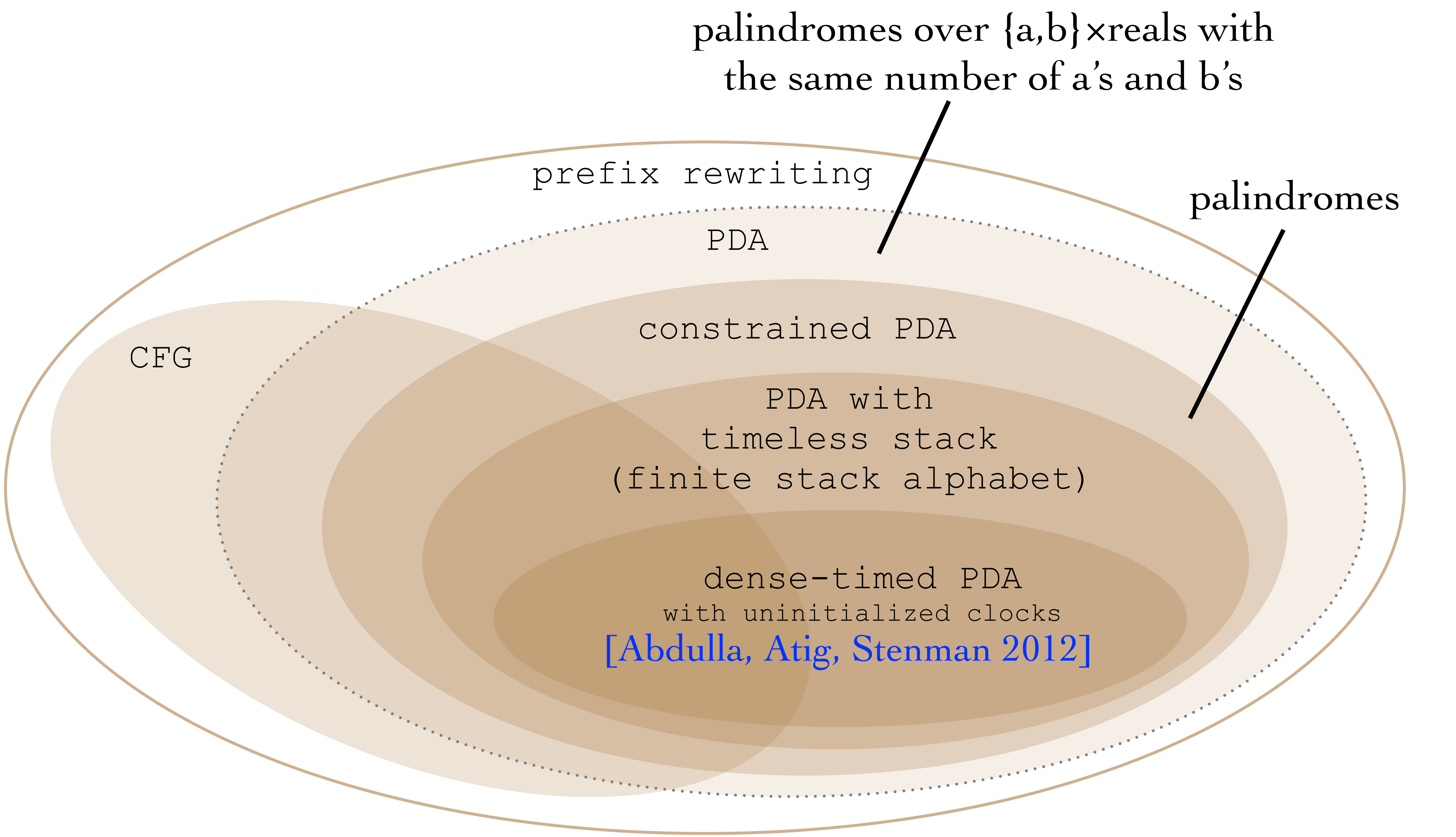
Expressiveness of FO-definable models

[Clemente, L. 2015]



Expressiveness of FO-definable models

[Clemente, L. 2015]



Constrained FO-definable PDA?

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- push $\subseteq$ Q $\times$ A $\times$ Q $\times$ S
- pop $\subseteq$ Q $\times$ S $\times$ A $\times$ Q
- I, F $\subseteq$ Q

definable in FO(<, +1)

Constrained FO-definable PDA?

- alphabet A
- states Q
- stack alphabet S
- push $\subseteq Q \times A \times Q \times S$
- pop $\subseteq Q \times S \times A \times Q$
- I, F $\subseteq Q$

orbit-finite

orbit-finite?

Constrained FO-definable PDA?

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- push $\subseteq Q \times A \times Q \times S$
- pop $\subseteq Q \times S \times A \times Q$
- $I, F \subseteq Q$

orbit-finite?

Too strong restriction! Span of transitions is bounded.

Constrained FO-definable PDA?

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- push $\subseteq Q \times A \times Q \times S$
- pop $\subseteq Q \times S \times A \times Q$
- I, F $\subseteq Q$

orbit-finite?

Too strong restriction! Span of transitions is bounded.

For instance, such PDA do not recognize palindromes over reals.

Constrained FO-definable PDA

- alphabet A
- states Q
- stack alphabet S
- push $\subseteq$ Q × A × Q × S
- pop $\subseteq$ Q × S × A × Q
- I, F $\subseteq$ Q

orbit-finite

definable in FO(<, +1)

Constrained FO-definable PDA

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- push $\subseteq Q \times A \times \underbrace{Q \times S}_{\text{orbit-finite}}$
- pop $\subseteq \underbrace{Q \times S}_{\text{orbit-finite}} \times A \times Q$
- I, F $\subseteq$ Q

definable in FO(<, +1)

Constrained FO-definable PDA

- alphabet A
- states Q
- stack alphabet S
- push $\subseteq$ Q × A × $\underbrace{Q \times S}_{\text{orbit-finite}}$
- pop $\subseteq$ $\underbrace{Q \times S}_{\text{orbit-finite}}$ × A × Q
- I, F $\subseteq$ Q

orbit-finite

definable in FO(<, +1)

Theorem 2: [Clemente, L. 2015]
The non-emptiness problem is in NEXPTIME.
For finite stack alphabet, EXPTIME-complete.

Constrained FO-definable PDA

- alphabet A
- states Q
- stack alphabet S

orbit-finite

- push $\subseteq Q \times A \times \underbrace{Q \times S}_{\text{orbit-finite}}$
- pop $\subseteq \underbrace{Q \times S}_{\text{orbit-finite}} \times A \times Q$
- I, F $\subseteq$ Q

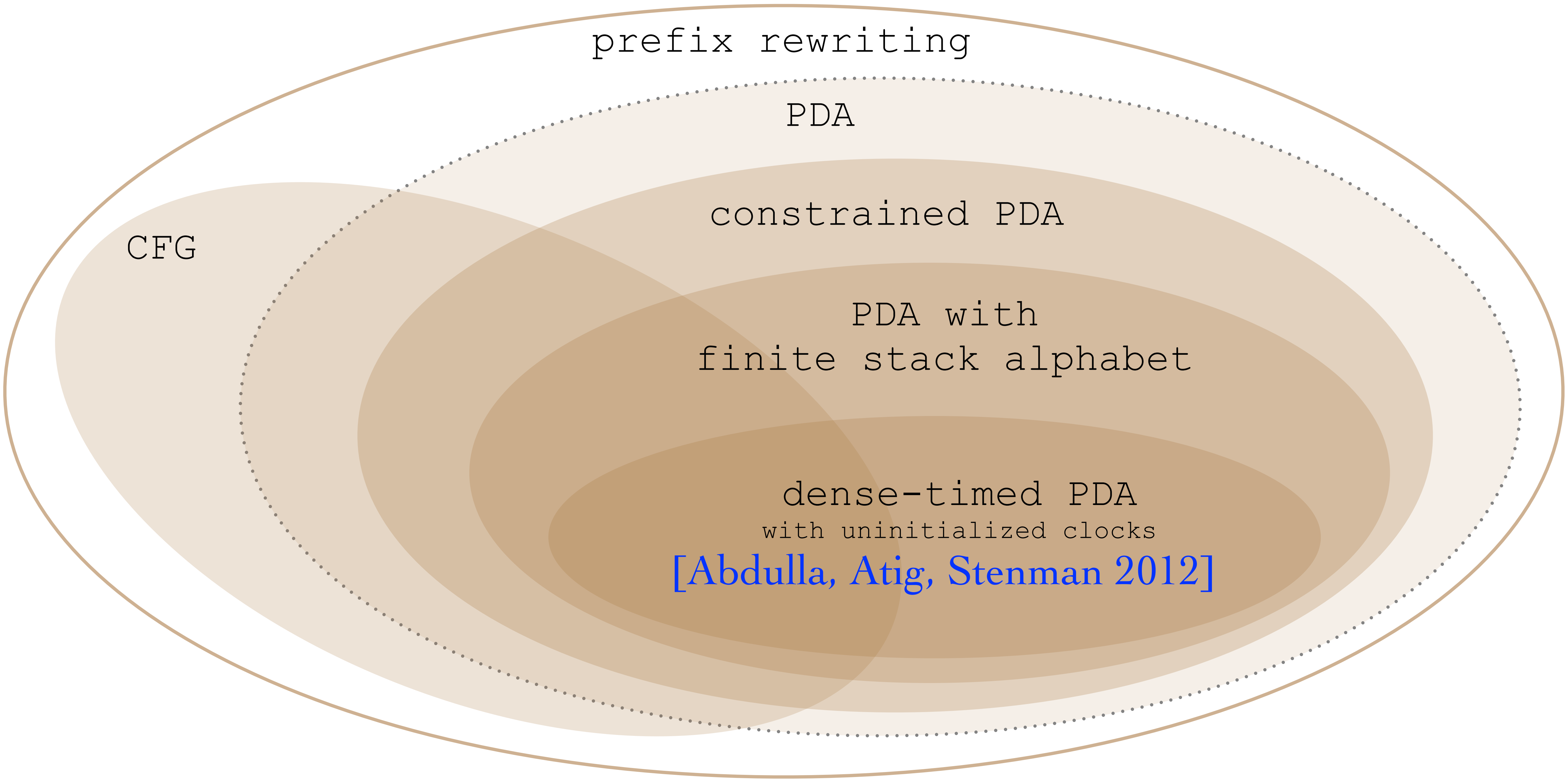
definable in FO(<, +1)

Theorem 2: [Clemente, L. 2015]
The non-emptiness problem is in NEXPTIME.
For finite stack alphabet, EXPTIME-complete.

Fact: The model subsumes dense-timed PDA with uninitialized clocks.

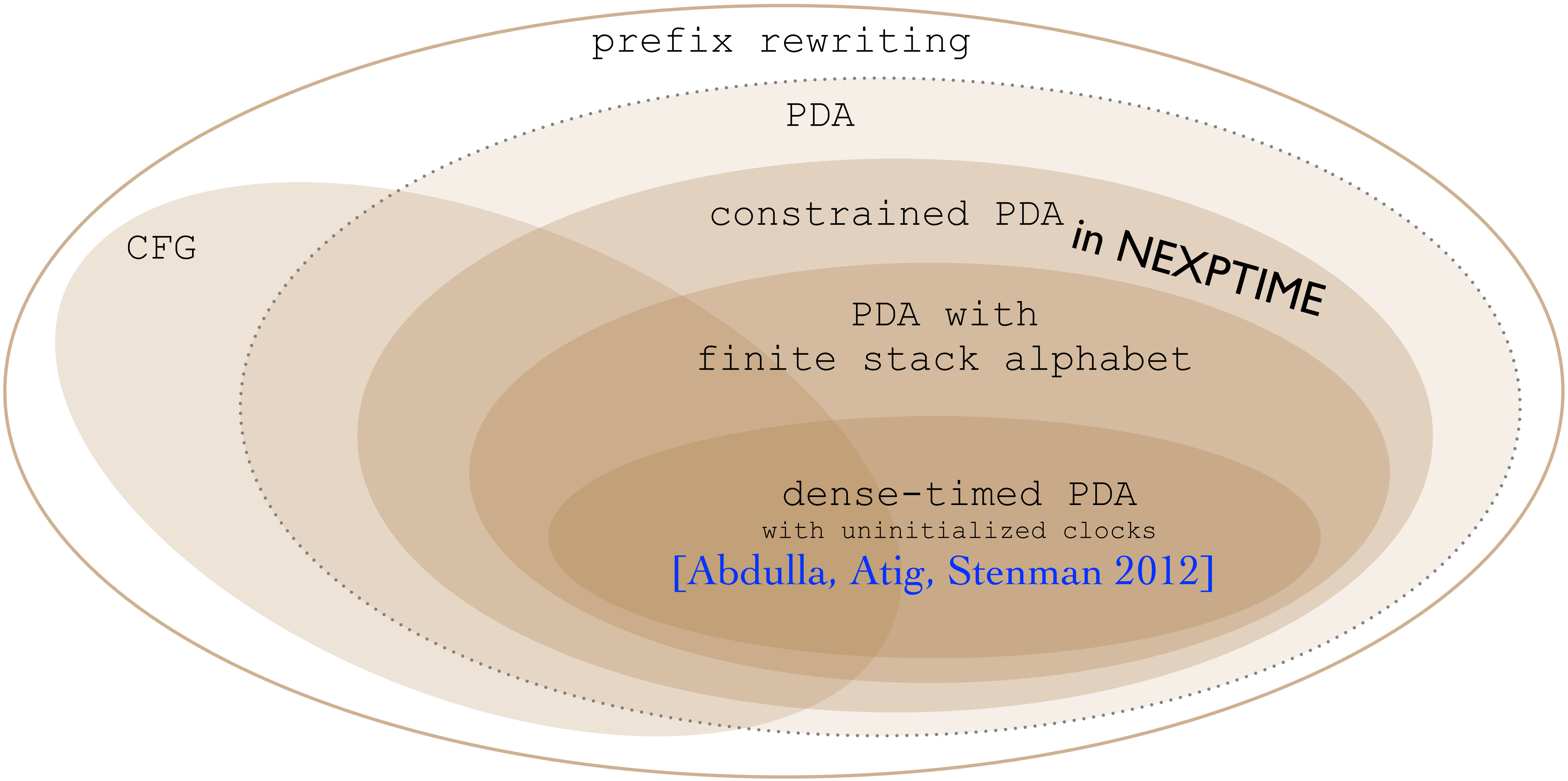
Complexity of non-emptiness

[Clemente, L. 2015]



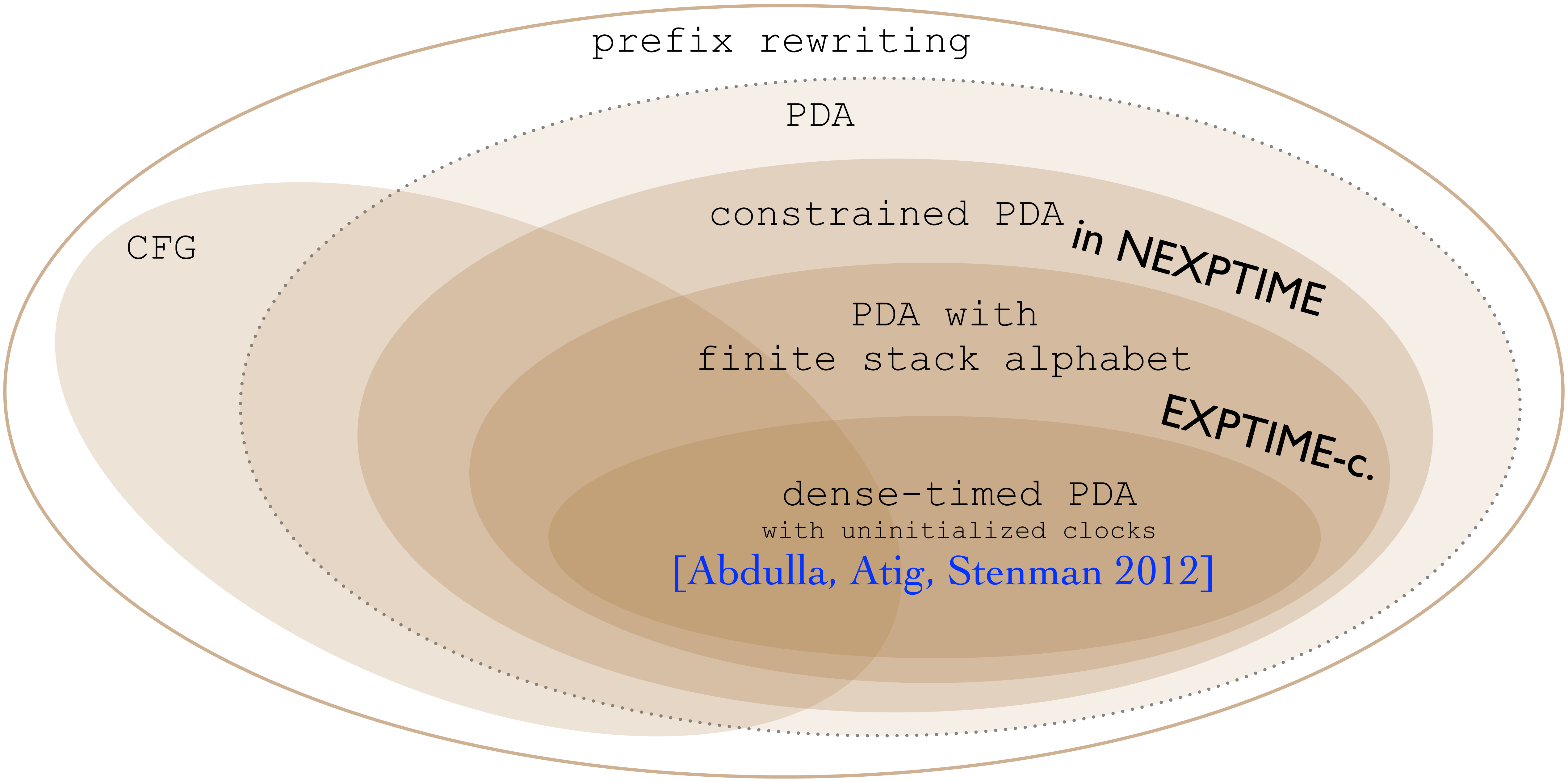
Complexity of non-emptiness

[Clemente, L. 2015]



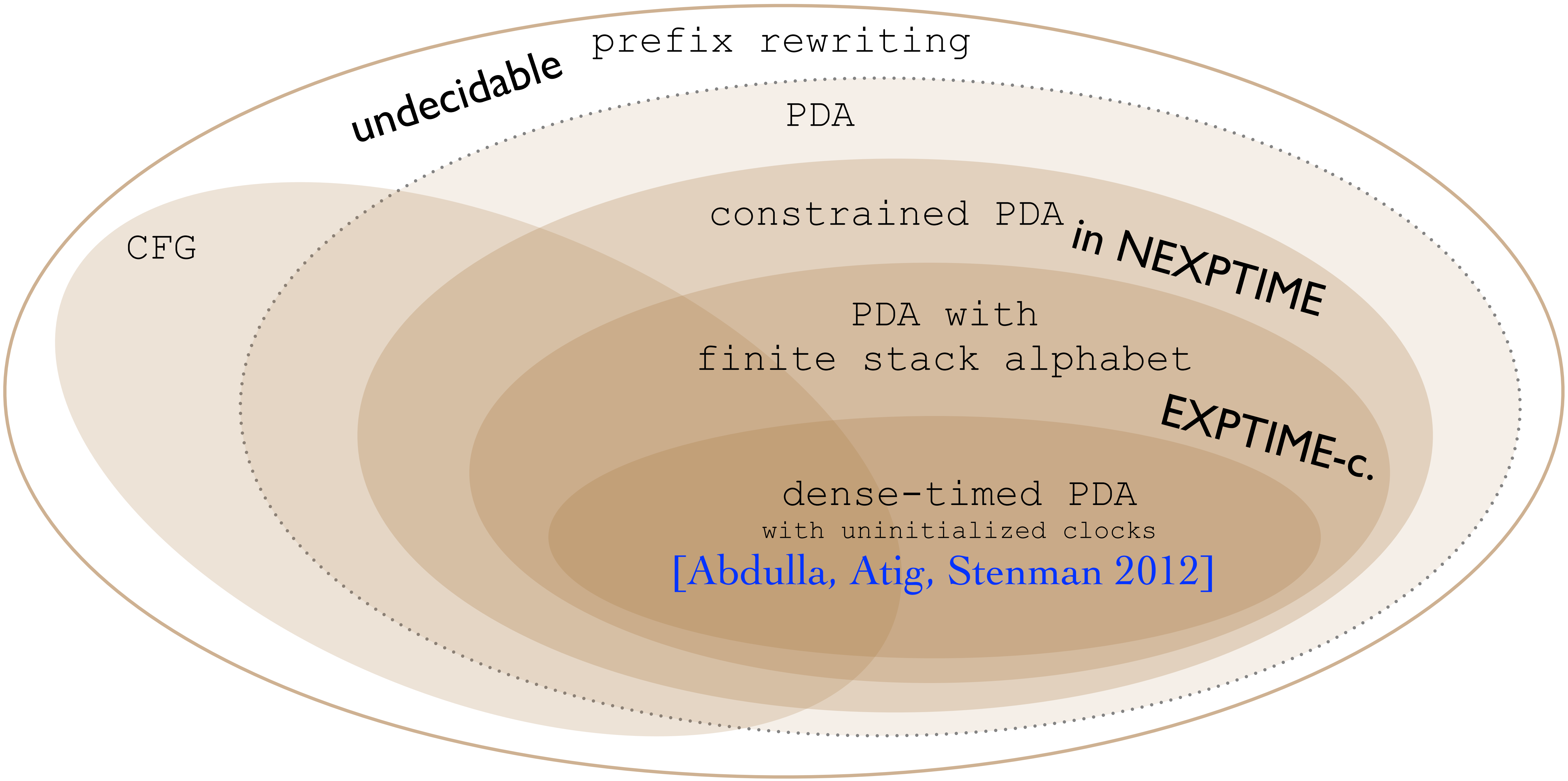
Complexity of non-emptiness

[Clemente, L. 2015]



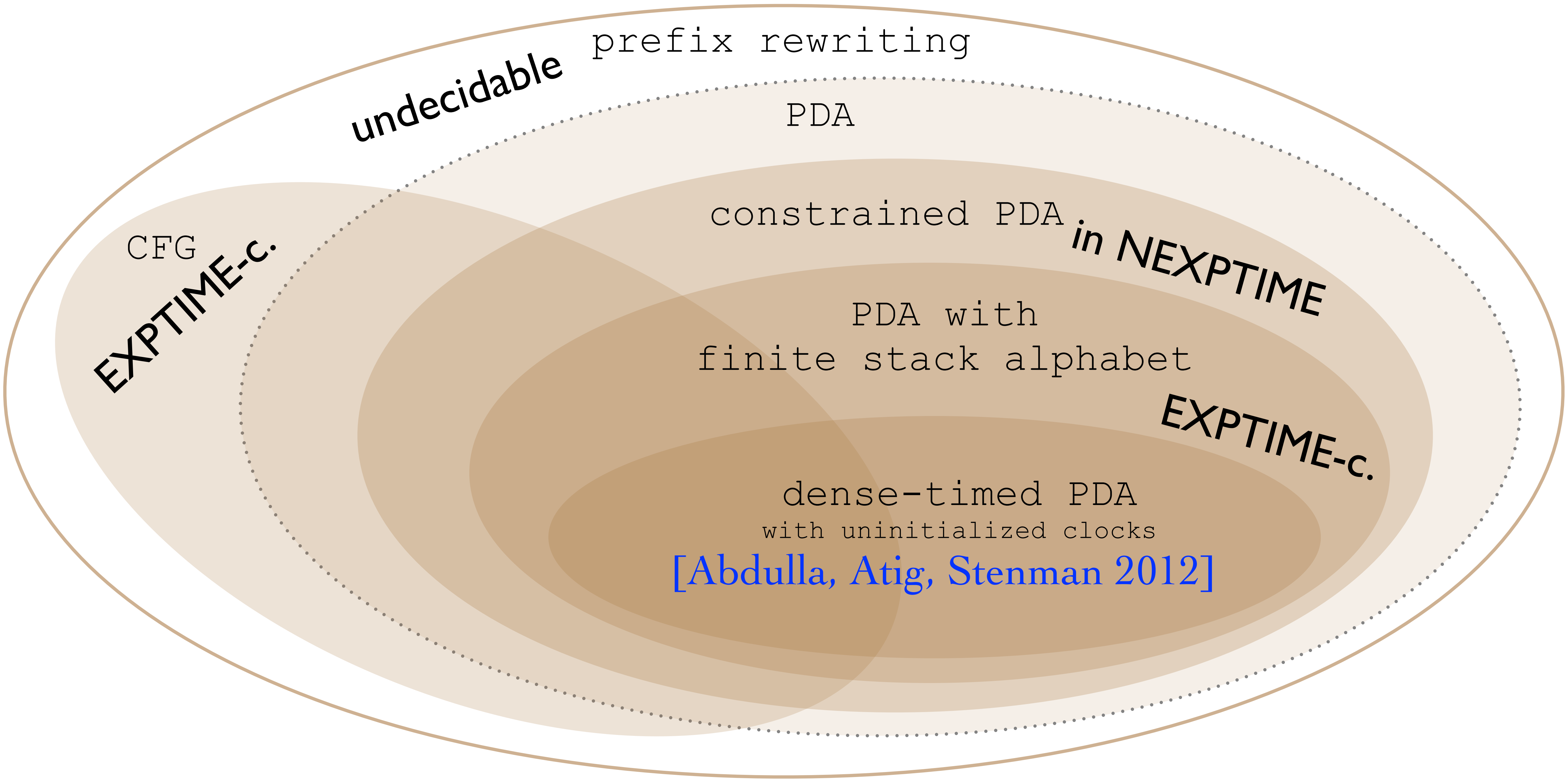
Complexity of non-emptiness

[Clemente, L. 2015]



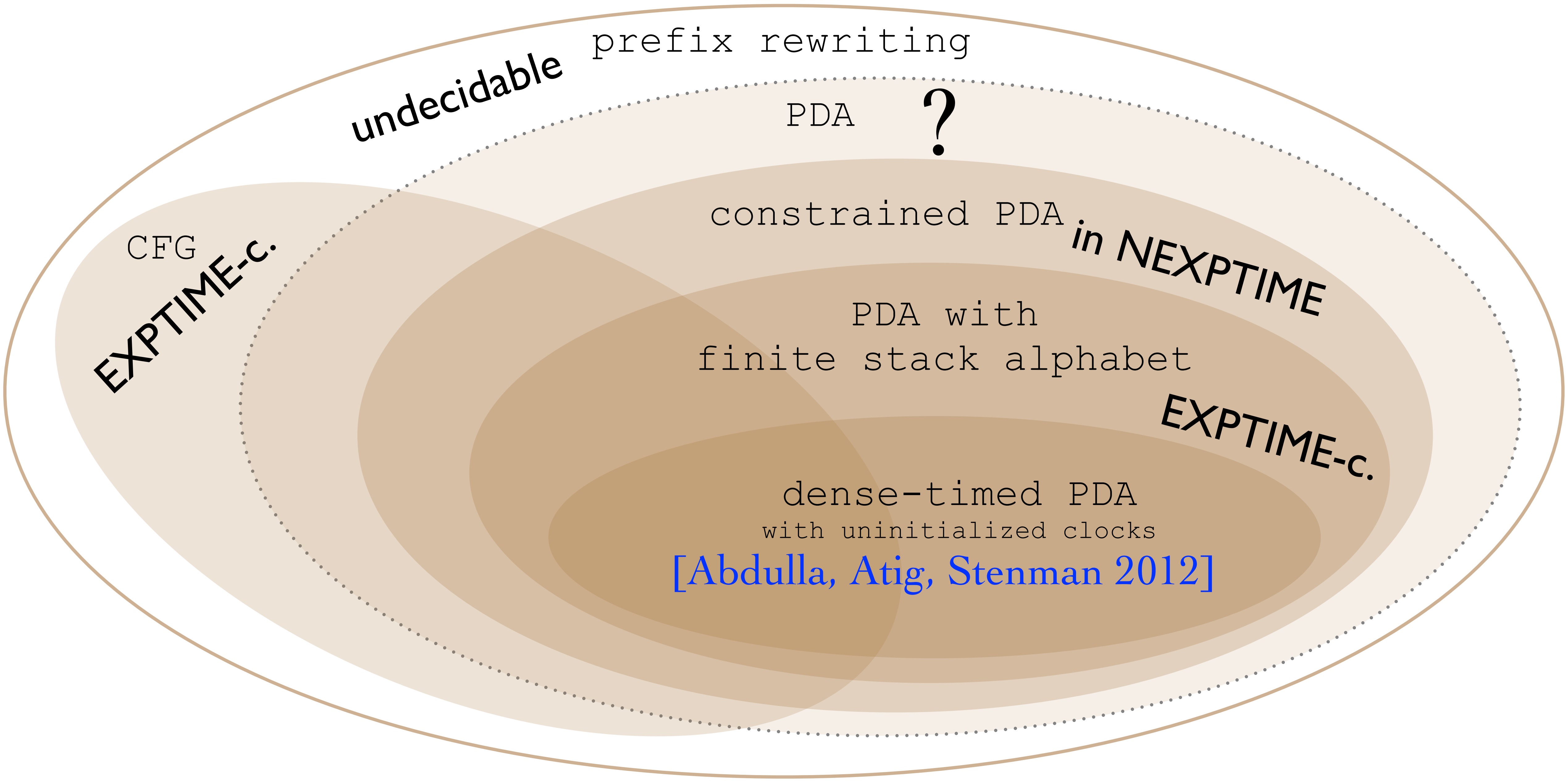
Complexity of non-emptiness

[Clemente, L. 2015]



Complexity of non-emptiness

[Clemente, L. 2015]



Plan

- Motivation
- FO-definable NFA
- FO-definable PDA
- The core problem: equations over sets of integers

The core problem

Systems of equations over sets of integers

$$\begin{cases} x_1 &= t_1 \\ x_2 &= t_2 \\ & \dots \\ x_n &= t_n \end{cases}$$

where right-hand sides use:

The core problem

Systems of equations over sets of integers

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

where right-hand sides use:

- constants {-1}, {0}, {1}

The core problem

Systems of equations over sets of integers

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \cdots & \\ x_n & = & t_n \end{cases}$$

where right-hand sides use:

- constants {-1}, {0}, {1}
- set union $\cup$

The core problem

Systems of equations over sets of integers

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \cdots & \\ x_n & = & t_n \end{cases}$$

where right-hand sides use:

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +

The core problem

Systems of equations over sets of integers

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \ldots & \\ x_n & = & t_n \end{cases}$$

where right-hand sides use:

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

The core problem

Systems of equations over sets of integers

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \cdots & \\ x_n & = & t_n \end{cases}$$

where right-hand sides use:

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

for instance:

$$\begin{cases} x_1 & = & \{0\} \cup x_2 + \{1\} \cup x_2 + \{-1\} \\ x_2 & = & x_1 + \{1\} \cup x_1 + \{-1\} \end{cases}$$

The core problem

Systems of equations over sets of integers

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \cdots & \\ x_n & = & t_n \end{cases}$$

where right-hand sides use:

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

for instance:

$$\begin{cases} x_1 & = & \{0\} \ \cup \ x_2 + \{1\} \ \cup \ x_2 + \{-1\} \\ x_2 & = & x_1 + \{1\} \ \cup \ x_1 + \{-1\} \end{cases}$$

What is the least solution with respect to inclusion?

The core problem - no intersections

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

decide, whether its least solution assigns a non-empty set to x_1?

The core problem - no intersections

Given a systems of equations

$$\begin{cases} x_1 &=& t_1 \\ x_2 &=& t_2 \\ & \dots & \\ x_n &=& t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

How to solve the problem in absence of intersections?

The core problem - no intersections

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

decide, whether its least solution assigns a non-empty set to x_1 ?

How to solve the problem in absence of intersections?

$$\begin{cases} x_1 & = & \{0\} \cup x_2 + \{1\} \cup x_2 + \{-1\} \\ x_2 & = & x_1 + \{1\} \cup x_1 + \{-1\} \end{cases}$$

The core problem - no intersections

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

decide, whether its least solution assigns a non-empty set to x_1?

How to solve the problem in absence of intersections?

$$\begin{cases} x_1 & = & \{0\} \cup x_2 + \{1\} \cup x_2 + \{-1\} \\ x_2 & = & x_1 + \{1\} \cup x_1 + \{-1\} \end{cases}$$

Decidable in P

The core problem - intersections

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

The core problem - intersections

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

decide, whether its least solution assigns a non-empty set to x_1?

The problem is undecidable for unlimited intersections.

[Jeż, Okhotin 2010]

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

decide, whether its least solution assigns a non-empty set to x_1?

What about limited intersections: _ ∩ I, for I a finite interval?

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

What about limited intersections: _ ∩ I, for I a finite interval?

$$\begin{cases} x_1 & = & \{0\} \;\cup\; x_2 + \{1\} \;\cup\; x_2 + \{-1\} \\ x_2 & = & (x_1 + \{1\} \;\cup\; x_1 + \{-1\}) \;\cap\; \{1\} \end{cases}$$

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 &= & t_1 \\ x_2 &= & t_2 \\ & \dots & \\ x_n &= & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

decide, whether its least solution assigns a non-empty set to x_1?

What about limited intersections: _ ∩ I, for I a finite interval?

$$\begin{cases} x_1 &= & \{0\} \ \cup \ x_2 + \{1\} \ \cup \ x_2 + \{-1\} \\ x_2 &= & x_1 + \{1\} \ \cup \ x_1 + \{-1\} \end{cases}$$

membership problem

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

What about limited intersections: _ ∩ I, for I a finite interval?

$$\begin{cases} x_1 & = & \{0\} \ \cup \ x_2 + \{1\} \ \cup \ x_2 + \{-1\} \\ x_2 & = & \{1\} \end{cases}$$

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 &= & t_1 \\ x_2 &= & t_2 \\ & \dots & \\ x_n &= & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

decide, whether its least solution assigns a non-empty set to x_1?

What about limited intersections: _ ∩ I, for I a finite interval?

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

decide, whether its least solution assigns a non-empty set to x_1 ?

What about limited intersections: _ ∩ I, for I a finite interval?

- NP-complete

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

What about limited intersections: _ ∩ I, for I a finite interval?

- NP-complete
- non-emptiness of constrained FO-definable PDA reduces to the core problem (with exponential blow-up)

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 &= & t_1 \\ x_2 &= & t_2 \\ & \dots & \\ x_n &= & t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union $\cup$
- point-wise addition +
- limited intersection $\cap$

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

What about _ ∩ I, for I an arbitrary interval?

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

decide, whether its least solution assigns a non-empty set to x_1?

What about _ ∩ I, for I an arbitrary interval?

- decidability status open!

The core problem - limited intersection

Given a systems of equations

$$\begin{cases} x_1 & = & t_1 \\ x_2 & = & t_2 \\ & \dots & \\ x_n & = & t_n \end{cases}$$

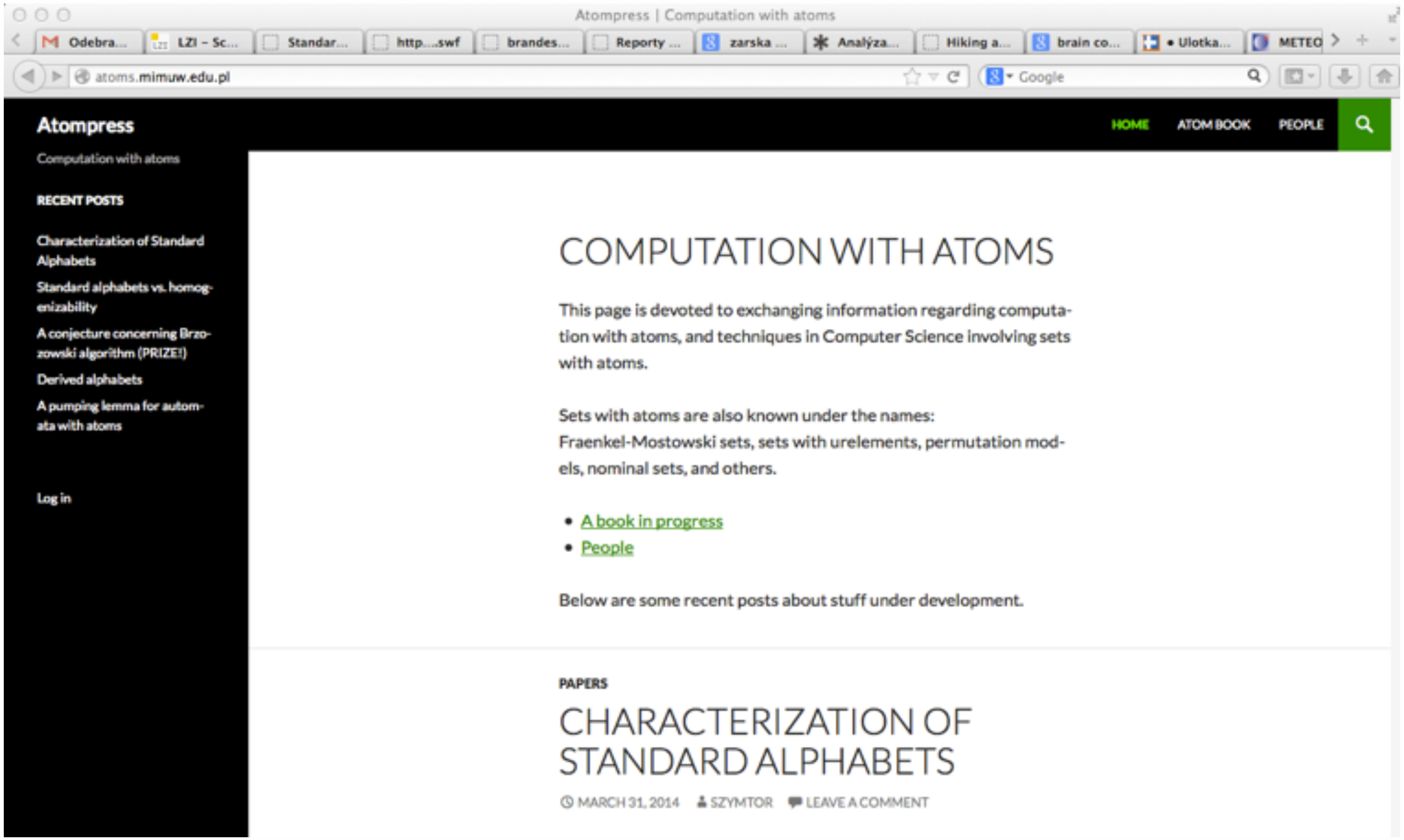
decide, whether its least solution assigns a non-empty set to x_1?

- constants {-1}, {0}, {1}
- set union ∪
- point-wise addition +
- limited intersection ∩

What about _ ∩ I, for I an arbitrary interval?

- decidability status open!
- non-emptiness of FO-definable PDA reduces to the core problem (with exponential blow-up)

Visit our blog...



questions?