Modelling time and recursion

L. Clemente, S. Lasota (University of Warsaw)
F. Mazowiecki, R. Lazić (University of Warwick)

Warsaw, July 2017

Summary

1. Modelling time: clocks vs registers.
2. Modelling recursion: timed pushdown automata.
3. Solution technique: reduction to 1-BVASS(±).

Clocks vs registers

In a nutshell:

- Clocks record the *difference* between events’ timestamps.
- Registers record the events’ timestamps themselves.

The two approaches are essentially equivalent*.

*with uninitialised clocks (preserves emptiness)

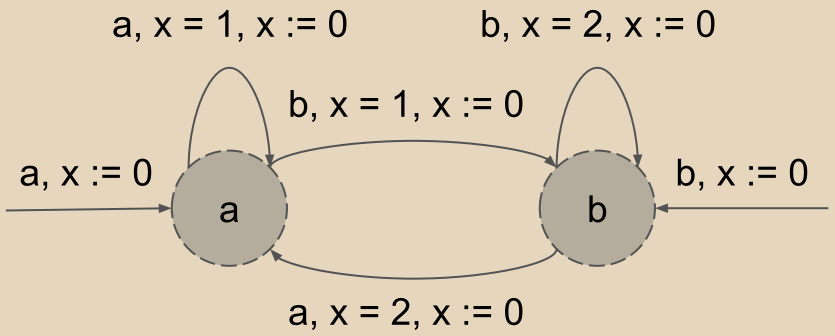
Clocks vs registers

Consider the timed language over $\{a, b\}^*$ “wait 1 sec after a and 2 sec after b”

Clocks vs registers

Consider the timed language over {a, b}* “wait 1 sec after a and 2 sec after b”

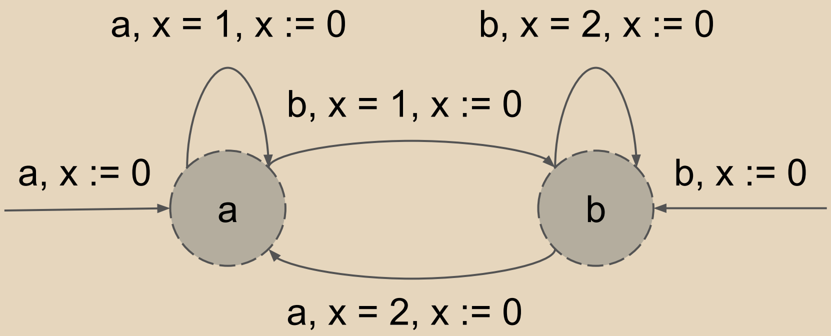
CLOCKS



Clocks vs registers

Consider the timed language over {a, b}* “wait 1 sec after a and 2 sec after b”

CLOCKS

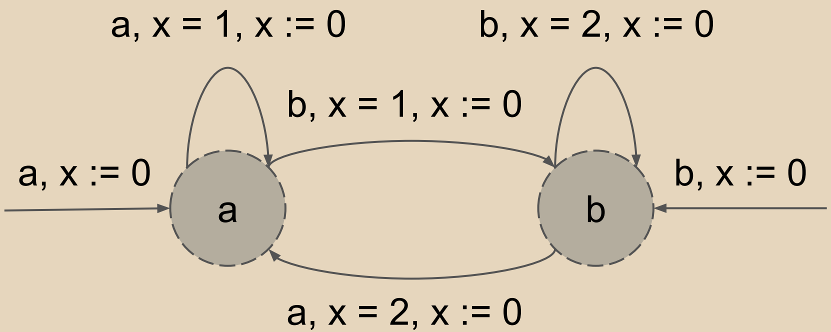


REGISTERS

Clocks vs registers

Consider the timed language over {a, b}* “wait 1 sec after a and 2 sec after b”

CLOCKS



t: current
timestamp

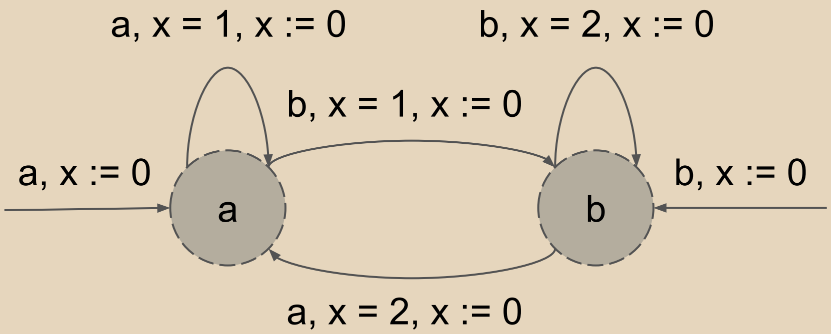
REGISTERS

x: current
x’: new

Clocks vs registers

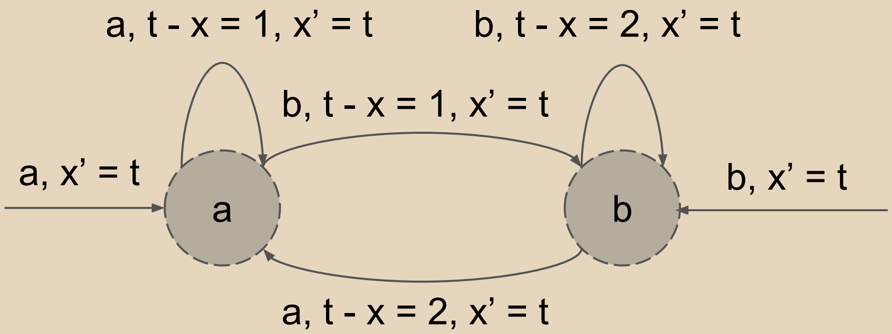
Consider the timed language over {a, b}* “wait 1 sec after a and 2 sec after b”

CLOCKS



t: current timestamp

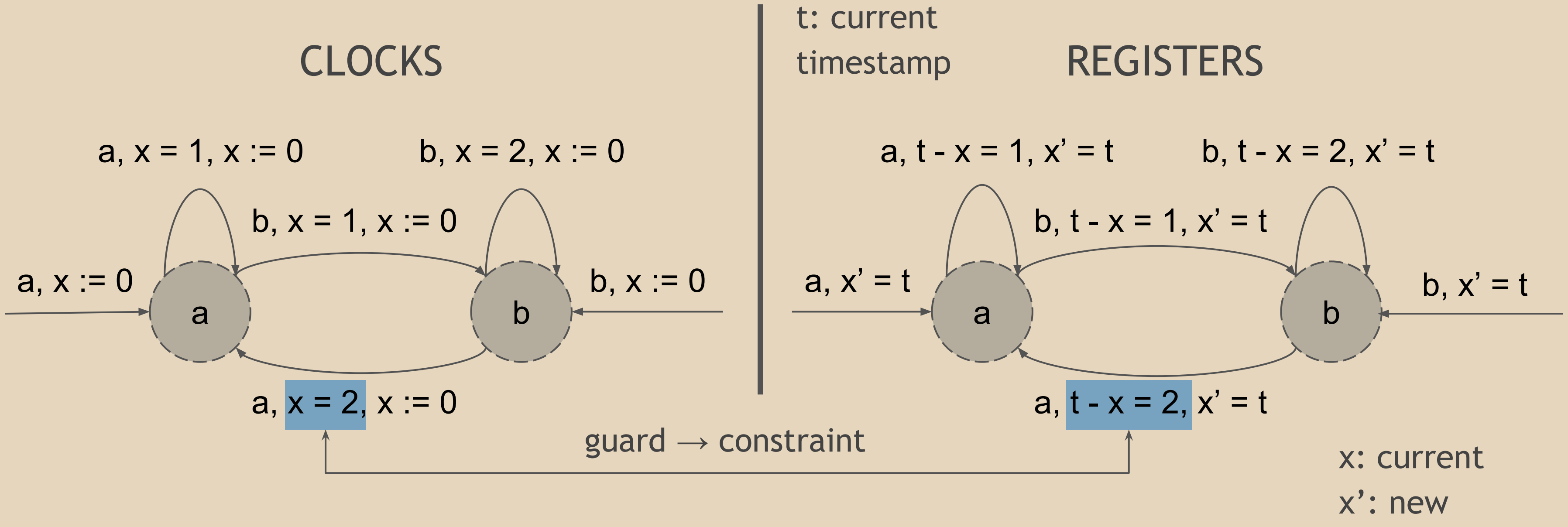
REGISTERS



x: current
x’: new

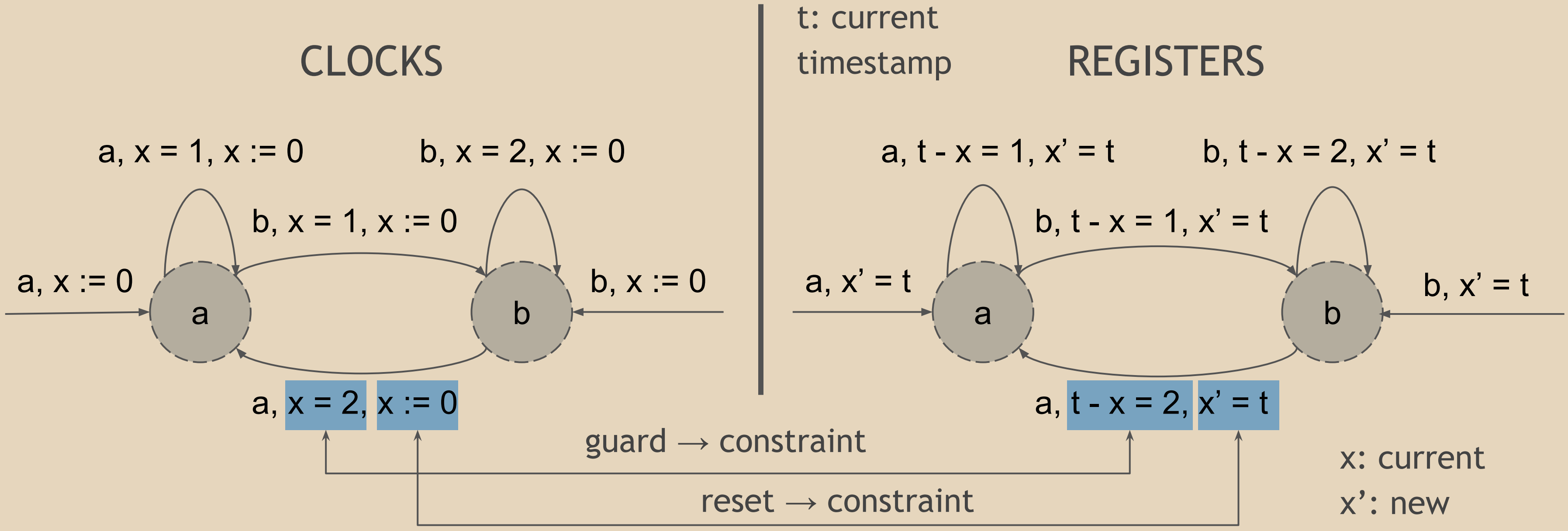
Clocks vs registers

Consider the timed language over {a, b}* “wait 1 sec after a and 2 sec after b”



Clocks vs registers

Consider the timed language over {a, b}* “wait 1 sec after a and 2 sec after b”



Timed register automata

Not a new concept:

- L. Fribourg, “A Closed-Form Evaluation for Extended Timed Automata”, techrep’98.
- H. Comon, Y. Jurski, “Timed Automata and the Theory of Real Numbers”, CONCUR’99.

Timed register automata

Not a new concept:

- L. Fribourg, “A Closed-Form Evaluation for Extended Timed Automata”, techrep’98.
- H. Comon, Y. Jurski, “Timed Automata and the Theory of Real Numbers”, CONCUR’99.

More recently by the Warsaw school of atoms (aka nominal sets, Fraenkel-Mostowski sets):

- D. Figueira, P. Hofman, S. Lasota, “Relating timed and register automata”, EXPRESS’10.
- M. Bojańczyk, S. Lasota, “A machine-independent characterization of timed languages”, ICALP’12.

Timed register automata

Not a new concept:

- L. Fribourg, “A Closed-Form Evaluation for Extended Timed Automata”, techrep’98.
- H. Comon, Y. Jurski, “Timed Automata and the Theory of Real Numbers”, CONCUR’99.

More recently by the Warsaw school of atoms (aka nominal sets, Fraenkel-Mostowski sets):

- D. Figueira, P. Hofman, S. Lasota, “Relating timed and register automata”, EXPRESS’10.
- M. Bojańczyk, S. Lasota, “A machine-independent characterization of timed languages”, ICALP’12.

FO-definable automata over

- $(\mathbb{N}, =)$: register automata [Kaminski, Francez TCS’94].

Timed register automata

Not a new concept:

- L. Fribourg, “A Closed-Form Evaluation for Extended Timed Automata”, techrep’98.
- H. Comon, Y. Jurski, “Timed Automata and the Theory of Real Numbers”, CONCUR’99.

More recently by the Warsaw school of atoms (aka nominal sets, Fraenkel-Mostowski sets):

- D. Figueira, P. Hofman, S. Lasota, “Relating timed and register automata”, EXPRESS’10.
- M. Bojańczyk, S. Lasota, “A machine-independent characterization of timed languages”, ICALP’12.

FO-definable automata over

- $(\mathbb{N}, =)$: register automata [Kaminski, Francez TCS’94].
- $(\mathbb{Z}, \leq, +1)$: discrete timed automata.

Timed register automata

Not a new concept:

- L. Fribourg, “A Closed-Form Evaluation for Extended Timed Automata”, techrep’98.
- H. Comon, Y. Jurski, “Timed Automata and the Theory of Real Numbers”, CONCUR’99.

More recently by the Warsaw school of atoms (aka nominal sets, Fraenkel-Mostowski sets):

- D. Figueira, P. Hofman, S. Lasota, “Relating timed and register automata”, EXPRESS’10.
- M. Bojańczyk, S. Lasota, “A machine-independent characterization of timed languages”, ICALP’12.

FO-definable automata over

- $(\mathbb{N}, =)$: register automata [Kaminski, Francez TCS’94].
- $(\mathbb{Z}, \leq, +1)$: discrete timed automata.
- $(\mathbb{R}, \leq, +1)$, $(\mathbb{Q}, \leq, +1)$: dense timed automata.

Timed register automata

Fix finitely many registers $X = \{x, y, \ldots\}$. A *timed register automaton* is a tuple

$$A = \langle Q, I, F, \Delta, \varphi(\delta) \rangle$$

where

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation.
- For every $\delta \in \Delta$, $\varphi(\delta)$ is a *constraint* using registers $X \cup X' \cup \{t\}$ over $(\mathbb{Q}, \leq, +1)$.

Timed register automata

Fix finitely many registers $X = \{x, y, \ldots\}$. A *timed register automaton* is a tuple

$$A = \langle Q, I, F, \Delta, \varphi(\delta) \rangle$$

where

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation.
- For every $\delta \in \Delta$, $\varphi(\delta)$ is a *constraint* using registers $X \cup X' \cup \{t\}$ over $(\mathbb{Q}, \leq, +1)$.
  - Atomic statements of the form: $x + 3 \leq y + 2$ with $x, y \in X \cup X' \cup \{t\}$.

Timed register automata

Fix finitely many registers $X = \{x, y, \ldots\}$. A *timed register automaton* is a tuple

$$A = \langle Q, I, F, \Delta, \varphi(\delta) \rangle$$

where

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation.
- For every $\delta \in \Delta$, $\varphi(\delta)$ is a *constraint* using registers $X \cup X' \cup \{t\}$ over $(\mathbb{Q}, \leq, +1)$.
  - Atomic statements of the form: $x + 3 \leq y + 2$ with $x, y \in X \cup X' \cup \{t\}$.
  - Boils down to conjunctions of $y - x \in I$, with I an interval in $\mathbb{Q} \cup \{+\infty, -\infty\}$.

Timed register automata vs Minsky

This model is too powerful. Can simulate 2-counter machines.

Timed register automata vs Minsky

This model is too powerful. Can simulate 2-counter machines.

Minsky

Let c, d be two counters.
Basic operations:

- c++.
- c--.
- c == 0.

Timed register automata vs Minsky

This model is too powerful. Can simulate 2-counter machines.

Minsky

Let c, d be two counters.
Basic operations:

- c++.
- c--.
- c == 0.

Registers

Let x, y, z be three registers over $(\mathbb{N}, \leq, +1)$.

Transformation: $c \rightarrow x - z$, $d \rightarrow y - z$

Timed register automata vs Minsky

This model is too powerful. Can simulate 2-counter machines.

Minsky

Let c, d be two counters.
Basic operations:

- c++.
- c--.
- c == 0.

Registers

Let x, y, z be three registers over $(\mathbb{N}, \leq, +1)$.

- $x' = x + 1$.
- $x' = x - 1 \wedge x' \geq z$.
- $x = z$, $x' = x$.

Transformation: $c \to x - z$, $d \to y - z$

Timed register automata vs Minsky

This model is too powerful. Can simulate 2-counter machines.

Minsky

Let c, d be two counters.
Basic operations:

- $c{+}{+}$.
- $c{-}{-}$.
- $c == 0$.

Registers

Let x, y, z be three registers over $(\mathbb{N}, \leq, +1)$.

- $x' = x + 1$.
- $x' = x - 1 \wedge x' \geq z$.
- $x = z, x' = x$.

registers x, y, z can have *unbounded* distances

Transformation: $c \to x - z$, $d \to y - z$

Timed register automata

Fix finitely many registers $X = \{x, y, \ldots\}$. A *timed register automaton* is a tuple

$$A = \langle Q, I, F, \Delta, \varphi(\delta), K \rangle$$

where

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation.
- For every $\delta \in \Sigma$, $\varphi(\delta)$ is a *constraint* using registers $X \cup X' \cup \{t\}$ over $(\mathbb{Q}, \leq, +1)$.

Timed register automata

Fix finitely many registers $X = \{x, y, \ldots\}$. A *timed register automaton* is a tuple

$$A = \langle Q, I, F, \Delta, \varphi(\delta), K \rangle$$

where

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation.
- For every $\delta \in \Sigma$, $\varphi(\delta)$ is a *constraint* using registers $X \cup X' \cup \{t\}$ over $(\mathbb{Q}, \leq, +1)$.
- $K \in \mathbb{N}$ is a *boundedness threshold*.
  - Valuations $\rho : X \mapsto \mathbb{Q}$ are restricted to satisfy: $\max |\rho(x) - \rho(y)| \leq K$.
  - ~ max constant in timed automata.

Timed register automata

Fix finitely many registers $X = \{x, y, \ldots\}$. A *timed register automaton* is a tuple

$$A = \langle Q, I, F, \Delta, \varphi(\delta), \boxed{K} \rangle$$

φ(δ) is still *unbounded*

where

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation.
- For every $\delta \in \Sigma$, $\varphi(\delta)$ is a *constraint* using registers $X \cup X' \cup \{t\}$ over $(\mathbb{Q}, \leq, +1)$.
- $K \in \mathbb{N}$ is a *boundedness threshold*.
  - Valuations $\rho : X \mapsto \mathbb{Q}$ are restricted to satisfy: $\max |\rho(x) - \rho(y)| \leq K$.
  - ~ max constant in timed automata.

Monotonicity of time

A timed register automaton recognises a language of *timed words*

$$L(A) \subseteq (\Sigma \times \mathbb{Q})^*.$$

There is no built-in notion of monotonicity of time.

Monotonicity of time

A timed register automaton recognises a language of *timed words*

$$L(A) \subseteq (\Sigma \times \mathbb{Q})^*.$$

There is no built-in notion of monotonicity of time.

Monotonic time *can be enforced* within the model:

- Add an extra register z.
- Add to every transition the constraint

$$z \leq t \land z' = t$$

Monotonicity of time

A timed register automaton recognises a language of *timed words*

$$L(A) \subseteq (\Sigma \times \mathbb{Q})^*.$$

There is no built-in notion of monotonicity of time.

Monotonic time *can be enforced* within the model:

- Add an extra register z.
- Add to every transition the constraint

$$z \leq t \wedge z' = t$$

check that the next timestamp is non-decreasing

Monotonicity of time

A timed register automaton recognises a language of *timed words*

$$L(A) \subseteq (\Sigma \times \mathbb{Q})^*.$$

There is no built-in notion of monotonicity of time.

Monotonic time *can be enforced* within the model:

- Add an extra register z.
- Add to every transition the constraint

check that the next timestamp is non-decreasing

save the timestamp

$$z \leq t \wedge z' = t$$

Pushdown automata + time

TA + (untimed) stack

Pushdown timed automata
[Bouajjani, Echahed, Robbana HS'94]

Pushdown automata + time

TA + (untimed) stack

Pushdown timed automata
[Bouajjani, Echahed, Robbana HS’94]

↓

Recursive timed automata
[Benerecetti, Minopoli, Peron TIME’10;
Trivedi, Wojtczak ATVA’10]

clocks on the stack are “frozen”

Pushdown automata + time

TA + (untimed) stack

Pushdown timed automata
[Bouajjani, Echahed, Robbana HS’94]

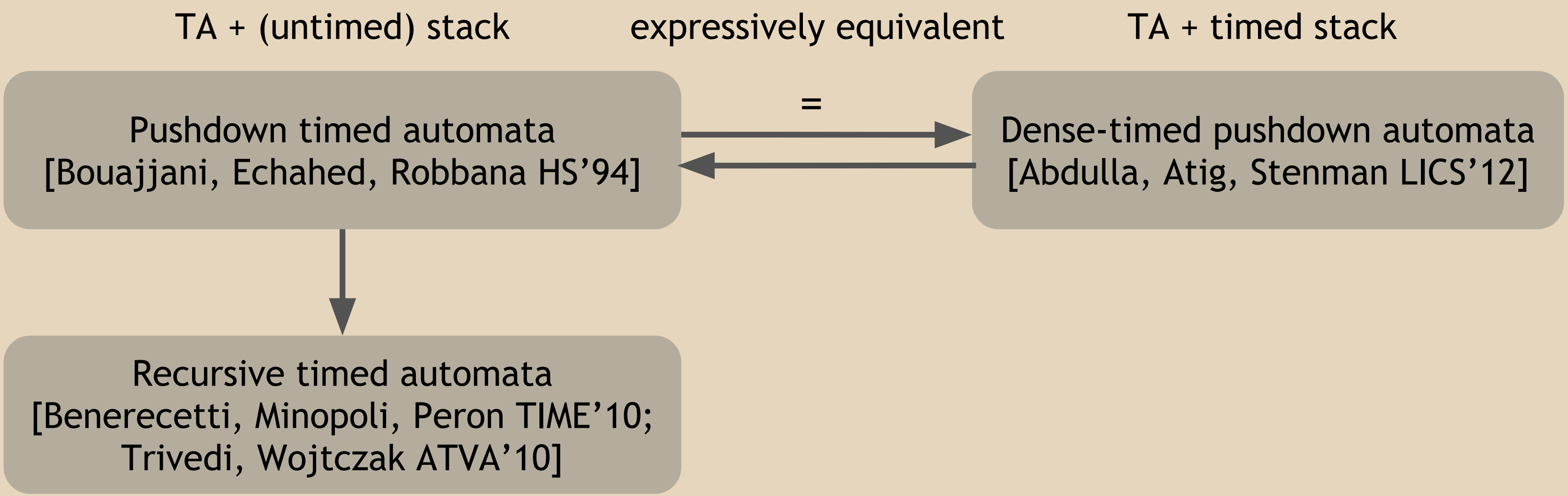
Recursive timed automata
[Benerecetti, Minopoli, Peron TIME’10;
Trivedi, Wojtczak ATVA’10]

clocks on the stack are “frozen”

TA + timed stack

Dense-timed pushdown automata
[Abdulla, Atig, Stenman LICS’12]

Pushdown automata + time



clocks on the stack are “frozen”

Pushdown automata + time

TA + (untimed) stack

expressively equivalent

TA + timed stack

Pushdown timed automata
[Bouajjani, Echahed, Robbana HS’94]

=

Dense-timed pushdown automata
[Abdulla, Atig, Stenman LICS’12]

Recursive timed automata
[Benerecetti, Minopoli, Peron TIME’10;
Trivedi, Wojtczak ATVA’10]

Timed register pushdown automata
[C, Lasota LICS’15;
C, Lasota, Lazić, Mazowiecki LICS’17]

clocks on the stack are “frozen”

registers on the stack

Dense-timed pushdown automata

In dtPDA [Abdulla, Atig, Stenman LICS’12]:

- Guards are of the form $x \in I$.
- Clocks can be pushed on the stack (w.l.o.g. initialised to zero).
- Clocks on the stack evolve at the same rate as control clocks.
- Clock x can be popped from the stack if it satisfies the *pop guard* $x \in I$.

Dense-timed pushdown automata

In dtPDA [Abdulla, Atig, Stenman LICS’12]:

- Guards are of the form $x \in I$.
- Clocks can be pushed on the stack (w.l.o.g. initialised to zero).
- Clocks on the stack evolve at the same rate as control clocks.
- Clock x can be popped from the stack if it satisfies the *pop guard* $x \in I$.

Limitations:

- No diagonal control-control clock constraints (this is not a limitation).

Dense-timed pushdown automata

In dtPDA [Abdulla, Atig, Stenman LICS'12]:

- Guards are of the form $x \in I$.
- Clocks can be pushed on the stack (w.l.o.g. initialised to zero).
- Clocks on the stack evolve at the same rate as control clocks.
- Clock x can be popped from the stack if it satisfies the *pop guard* $x \in I$.

Limitations:

- No diagonal control-control clock constraints (this is not a limitation).
- No diagonal control-stack push clock constraints (unknown).
- No diagonal control-stack pop clock constraints (this is not a limitation).

Semantic collapse of dtPDA

Very strong collapse result:

Theorem [CL LICS’15]. dtPDA of [AAS LICS’12] recognise the same class of timed languages as pushdown timed automata of [BER HS’94].

Semantic collapse of dtPDA

Very strong collapse result:

Theorem [CL LICS’15]. dtPDA of [AAS LICS’12] recognise the same class of timed languages as pushdown timed automata of [BER HS’94].

In other words, the stack can be *untimed*.

Semantic collapse of dtPDA

Very strong collapse result:

Theorem [CL LICS’15]. dtPDA of [AAS LICS’12] recognise the same class of timed languages as pushdown timed automata of [BER HS’94].

In other words, the stack can be *untimed*.

Intuition:

- Time is monotone + stack LIFO policy
  ⇒ it suffices to keep track of finitely many pop constraints in the state
  ⇒ pop guards can be eliminated while preserving the timed language

Semantic collapse of dtPDA

Pop guards of the form $x \in [2, 3]$ + time is monotone + stack LIFO policy

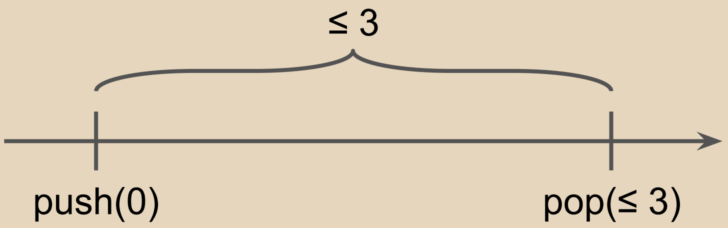
Upper bound constraints:
old subsumes new

Semantic collapse of dtPDA

Pop guards of the form x ∈ [2, 3] + time is monotone + stack LIFO policy

Upper bound constraints:

old subsumes new

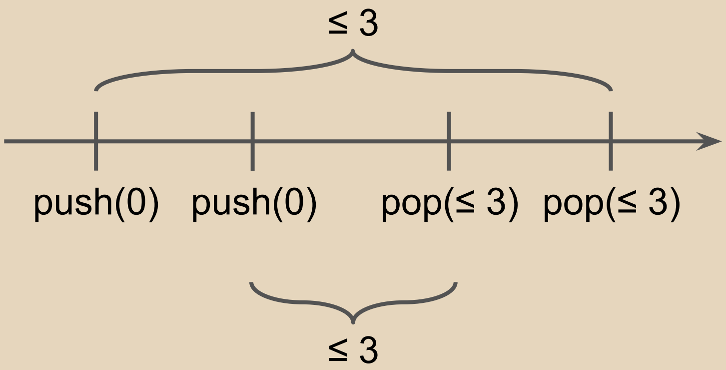


Semantic collapse of dtPDA

Pop guards of the form x ∈ [2, 3] + time is monotone + stack LIFO policy

Upper bound constraints:

old subsumes new

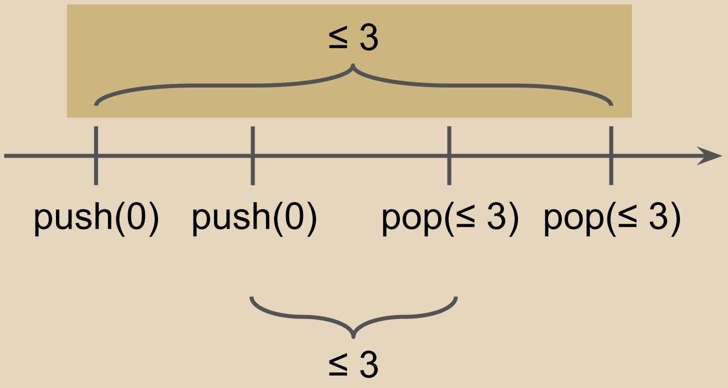


Semantic collapse of dtPDA

Pop guards of the form x ∈ [2, 3] + time is monotone + stack LIFO policy

Upper bound constraints:

old subsumes new

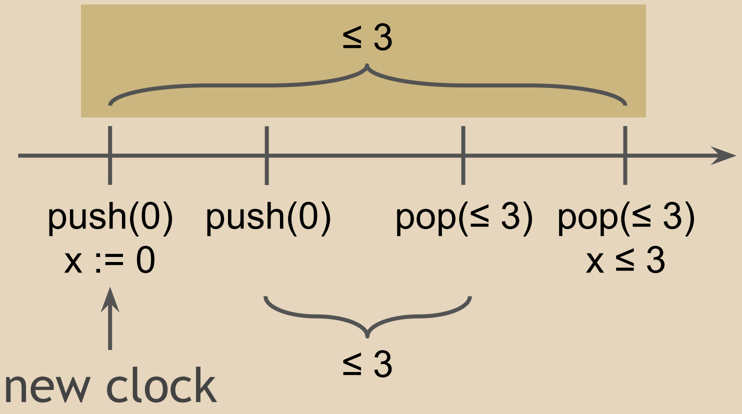


Semantic collapse of dtPDA

Pop guards of the form x ∈ [2, 3] + time is monotone + stack LIFO policy

Upper bound constraints:

old subsumes new

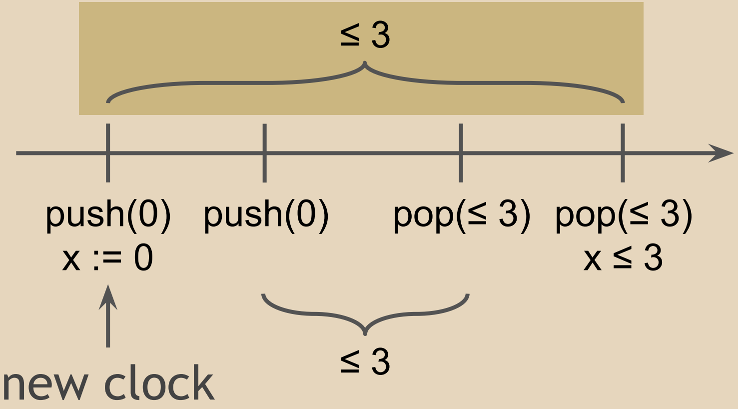


Semantic collapse of dtPDA

Pop guards of the form x ∈ [2, 3] + time is monotone + stack LIFO policy

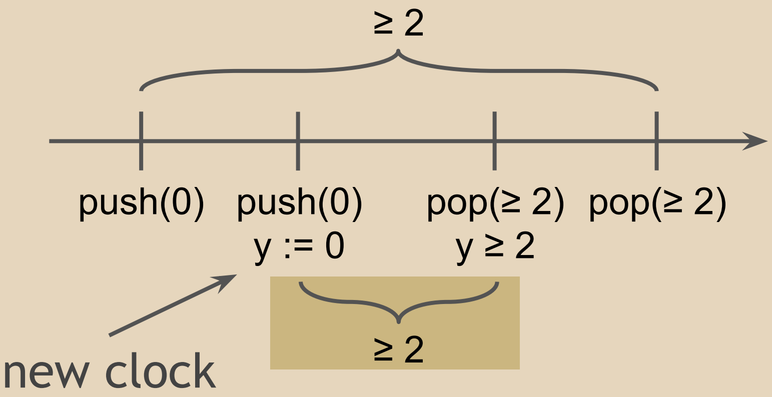
Upper bound constraints:

old subsumes new



Lower bound constraints:

new subsumes old



Semantic collapse of dtPDA

Very strong collapse result:

Theorem [CL LICS'15]. dtPDA of [AAS LICS'12] recognise the same class of timed languages as pushdown timed automata of [BER HS'94].

Consequences:

- dtPDA are expressively equivalent to TA + untimed stack (PDTA).

Semantic collapse of dtPDA

Very strong collapse result:

Theorem [CL LICS'15]. dtPDA of [AAS LICS'12] recognise the same class of timed languages as pushdown timed automata of [BER HS'94].

Consequences:

- dtPDA are expressively equivalent to TA + untimed stack (PDTA).

Complexity:

- Add linearly many clocks and exponentially many control locations.
- Emptiness of PDTA is exponential in the number of clocks and polynomial in the number of control locations $\Rightarrow$ emptiness of dtPDA is in EXPTIME.

Follow-ups to dtPDA

Abdulla, Atig, Stenman, “Zenoness for Timed Pushdown Automata”, INFINITY’13.
Abdulla, Atig, Stenman, “Computing Optimal Reachability Costs in Priced Dense-Timed Pushdown Automata, LATA’14.

Follow-ups to dtPDA

Abdulla, Atig, Stenman, “Zenoness for Timed Pushdown Automata”, INFINITY’13.
Abdulla, Atig, Stenman, “Computing Optimal Reachability Costs in Priced Dense-Timed Pushdown Automata, LATA’14.

Uezato, Minamide, “Synchronized Recursive Timed Automata”, LPAR’15. (diagonal + fractional constraints)

Li, Cai, Ogawa, Yuen, “Nested timed automata”, FORMATS’13. (reduction to dtPDA)
Cai, Ogawa, “Well-structured pushdown system: Case of Dense Timed Pushdown Automata”, FLOPS’14.

Krishna, Manasa, Trivedi, “Reachability Games on Recursive Hybrid Automata”, TIME’15.

Follow-ups to dtPDA

Abdulla, Atig, Stenman, “Zenoness for Timed Pushdown Automata”, INFINITY’13.
Abdulla, Atig, Stenman, “Computing Optimal Reachability Costs in Priced Dense-Timed Pushdown Automata, LATA’14.

Uezato, Minamide, “Synchronized Recursive Timed Automata”, LPAR’15. (diagonal + fractional constraints)

Li, Cai, Ogawa, Yuen, “Nested timed automata”, FORMATS’13. (reduction to dtPDA)
Cai, Ogawa, “Well-structured pushdown system: Case of Dense Timed Pushdown Automata”, FLOPS’14.

Krishna, Manasa, Trivedi, “Reachability Games on Recursive Hybrid Automata”, TIME’15.

Droste, Perevoshchikov, “A Logical Characterization of Timed Pushdown Languages”, CSR’15. (the logic collapses)

Follow-ups to dtPDA

Abdulla, Atig, Stenman, “Zenoness for Timed Pushdown Automata”, INFINITY’13.
Abdulla, Atig, Stenman, “Computing Optimal Reachability Costs in Priced Dense-Timed Pushdown Automata, LATA’14.

Uezato, Minamide, “Synchronized Recursive Timed Automata”, LPAR’15. (diagonal + fractional constraints)

Li, Cai, Ogawa, Yuen, “Nested timed automata”, FORMATS’13. (reduction to dtPDA)
Cai, Ogawa, “Well-structured pushdown system: Case of Dense Timed Pushdown Automata”, FLOPS’14.

Krishna, Manasa, Trivedi, “Reachability Games on Recursive Hybrid Automata”, TIME’15.

Droste, Perevoshchikov, “A Logical Characterization of Timed Pushdown Languages”, CSR’15. (the logic collapses)
Bhave, Dave, Krishna, Phawade, Trivedi, “A Logical Characterization for Dense-time Visibly Pushdown Automata”, LATA’16.
Bhave, Dave, Krishna, Phawade, Trivedi, “A Perfect Class of Context-Sensitive Timed Languages”, DLT’16. (multitstack)

Follow-ups to dtPDA

Abdulla, Atig, Stenman, “Zenoness for Timed Pushdown Automata”, INFINITY’13.
Abdulla, Atig, Stenman, “Computing Optimal Reachability Costs in Priced Dense-Timed Pushdown Automata, LATA’14.

Uezato, Minamide, “Synchronized Recursive Timed Automata”, LPAR’15. (diagonal + fractional constraints)

Li, Cai, Ogawa, Yuen, “Nested timed automata”, FORMATS’13. (reduction to dtPDA)
Cai, Ogawa, “Well-structured pushdown system: Case of Dense Timed Pushdown Automata”, FLOPS’14.

Krishna, Manasa, Trivedi, “Reachability Games on Recursive Hybrid Automata”, TIME’15.

Droste, Perevoshchikov, “A Logical Characterization of Timed Pushdown Languages”, CSR’15. (the logic collapses)
Bhave, Dave, Krishna, Phawade, Trivedi, “A Logical Characterization for Dense-time Visibly Pushdown Automata”, LATA’16.
Bhave, Dave, Krishna, Phawade, Trivedi, “A Perfect Class of Context-Sensitive Timed Languages”, DLT’16. (multitstack)

Akshay, Gastin, Krishna, “Analyzing Timed Systems Using Tree Automata”, CONCUR’16. (tree-width approach)

(Non)Example

Example from [BDKPT LATA'16] about logical characterisation of dtVPA.

L = words of the form $a^n\ b\ c^n\ d$ with n ≥ 0 s.t.

- first c comes after 1 time unit after last a
- first a and last c are 2 time units apart
- every other matching a and c are in (1, 2)

(Non)Example

Example from [BDKPT LATA'16] about logical characterisation of dtVPA.

L = words of the form $a^n\ b\ c^n\ d$ with n ≥ 0 s.t.

- first c comes after 1 time unit after last a [1 clock],
- first a and last c are 2 time units apart [1 clock],
- every other matching a and c are in (1, 2) [2 clocks].

(Non)Example

Example from [BDKPT LATA'16] about logical characterisation of dtVPA.

L = words of the form $a^n\ b\ c^n\ d$ with n ≥ 0 s.t.

- first c comes after 1 time unit after last a [1 clock],
- first a and last c are 2 time units apart [1 clock],
- every other matching a and c are in (1, 2) [2 clocks].

We do not need a timed stack to recognise this language (4 clocks suffice).
In fact, they show that the stack can be untimed in the spirit of [CL LICS'15].

Example

Consider the language of *timed palindromes*

$$L = \{ w\, w^R \mid w \in (\Sigma \times \mathbb{Q})^* \}$$

- It requires a truly timed stack.
- Cannot be expressed in any of the previous models.

Example

Consider the language of *timed palindromes*

$$L = \{ w\, w^R \mid w \in (\Sigma \times \mathbb{Q})^* \}$$

- It requires a truly timed stack.
- Cannot be expressed in any of the previous models.
- It is a non-monotone language.
    - Can be made monotone by requiring palindromicity only for the fractional values.

Timed register pushdown automata

Fix a finite set of registers X, Y, input alphabet Σ, and stack alphabet Γ.
A *timed register pushdown automaton* (trPDA) is a tuple

$$A = \langle Q, I, F, PUSH, POP, \{ \varphi(\delta) \mid \delta \in PUSH \cup POP \}, K \rangle \text{where}$$

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $PUSH, POP \subseteq Q \times \Sigma \times Q \times \Gamma$ is the transition relation.

Timed register pushdown automata

Fix a finite set of registers X, Y, input alphabet Σ, and stack alphabet Γ.

A *timed register pushdown automaton* (trPDA) is a tuple

$$A = \langle Q, I, F, PUSH, POP, \{ \varphi(\delta) \mid \delta \in PUSH \cup POP \}, K \rangle \text{ where}$$

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $PUSH, POP \subseteq Q \times \Sigma \times Q \times \Gamma$ is the transition relation.
- For every $\delta \in PUSH \cup POP$, $\varphi(\delta)$ is a *constraint* over $(\mathbb{Q},\leq,+1)$ using registers

$$X \cup X' \cup Y \cup \{ t \}.$$

- $K \in \mathbb{N}$ is a *boundedness threshold* for state and stack registers.

Timed register pushdown automata

Fix a finite set of registers X, Y, input alphabet Σ, and stack alphabet Γ.

A *timed register pushdown automaton* (trPDA) is a tuple

$$A = \langle Q, I, F, PUSH, POP, \{ \varphi(\delta) \mid \delta \in PUSH \cup POP \}, K \rangle \text{ where}$$

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $PUSH, POP \subseteq Q \times \Sigma \times Q \times \Gamma$ is the transition relation.
- For every $\delta \in PUSH \cup POP$, $\varphi(\delta)$ is a *constraint* over $(\mathbb{Q}, \leq, +1)$ using registers

stack registers

$$X \cup X' \cup Y \cup \{ t \}.$$

- $K \in \mathbb{N}$ is a *boundedness threshold* for state and stack registers.

Timed register pushdown automata

Fix a finite set of registers X, Y, input alphabet Σ, and stack alphabet Γ.
A *timed register pushdown automaton* (trPDA) is a tuple

$$A = \langle Q, I, F, PUSH, POP, \{ \varphi(\delta) \mid \delta \in PUSH \cup POP \}, K \rangle \text{ where}$$

- Q is a finite set of control states, with $I, F \subseteq Q$ the initial, final ones, resp.
- $PUSH, POP \subseteq Q \times \Sigma \times Q \times \Gamma$ is the transition relation.
- For every $\delta \in PUSH \cup POP$, $\varphi(\delta)$ is a *constraint* over $(\mathbb{Q}, \leq, +1)$ using registers

stack registers

$$X \cup X' \cup \boxed{Y} \cup \{ t \}.$$

$\varphi(\delta)$ is *unbounded*

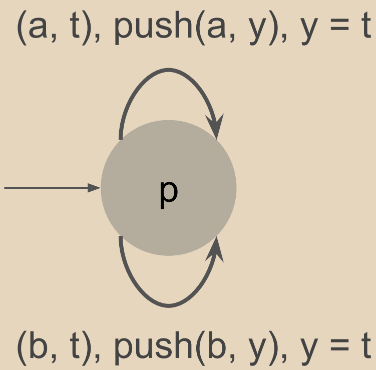
- $K \in \mathbb{N}$ is a *boundedness threshold* for state and stack registers.

Example (1)

Timed palindromes over $\Sigma = \{a, b\}$: $L = \{ w\, w^R \mid w \in (\Sigma \times \mathbb{Q})^* \}$.

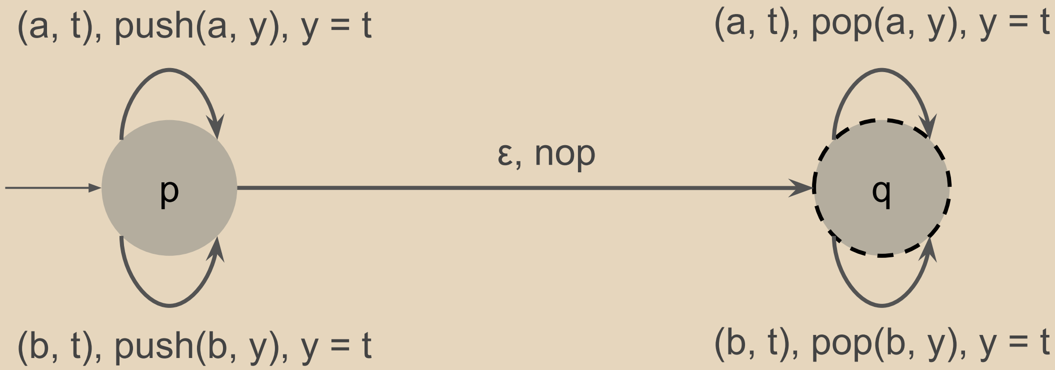
Example (1)

Timed palindromes over $\Sigma = \{a, b\}$: $L = \{ w\, w^R \mid w \in (\Sigma \times \mathbb{Q})^* \}$.



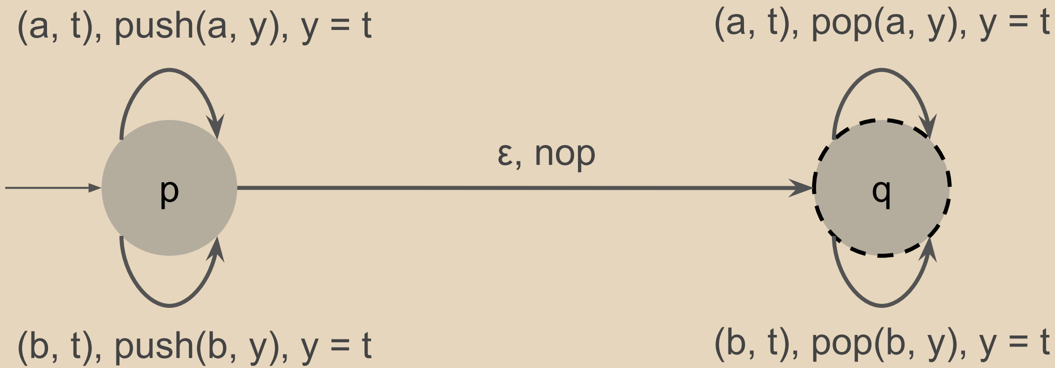
Example (1)

Timed palindromes over $\Sigma = \{a, b\}$: $L = \{ w\, w^R \mid w \in (\Sigma \times \mathbb{Q})^* \}$.



Example (1)

Timed palindromes over $\Sigma = \{a, b\}$: $L = \{ w\, w\text{^}R \mid w \in (\Sigma \times \mathbb{Q})^* \}$.



The untiming projection of L is a context-free language.

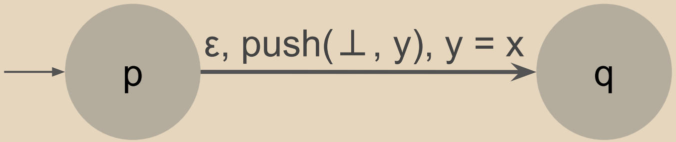
Example (2)

Untimed palindromes *with the same number of a’s and b’s*.

***not* a context-free language**

Example (2)

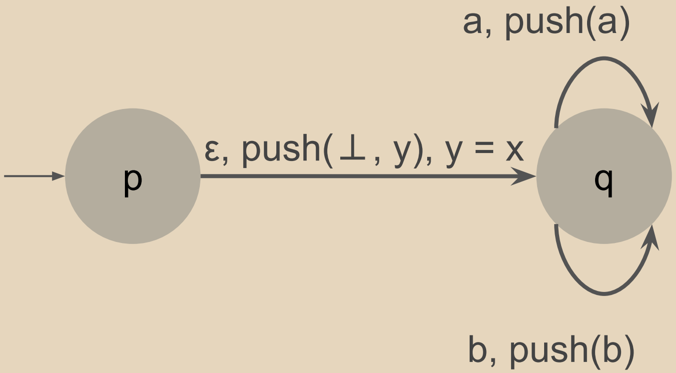
Untimed palindromes *with the same number of a's and b's*.



not **a context-free language**

Example (2)

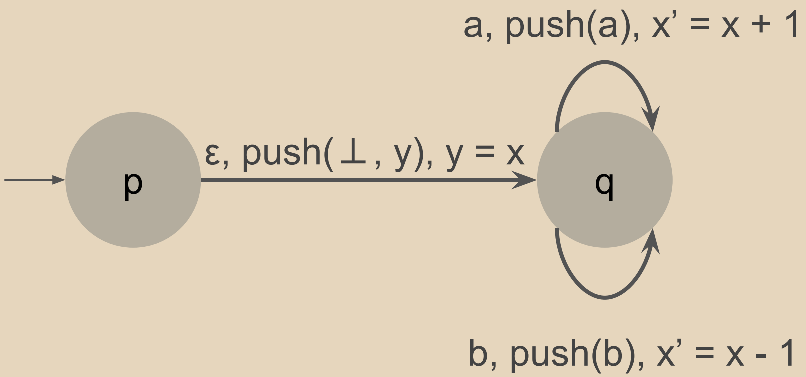
Untimed palindromes *with the same number of a's and b's*.



not **a context-free language**

Example (2)

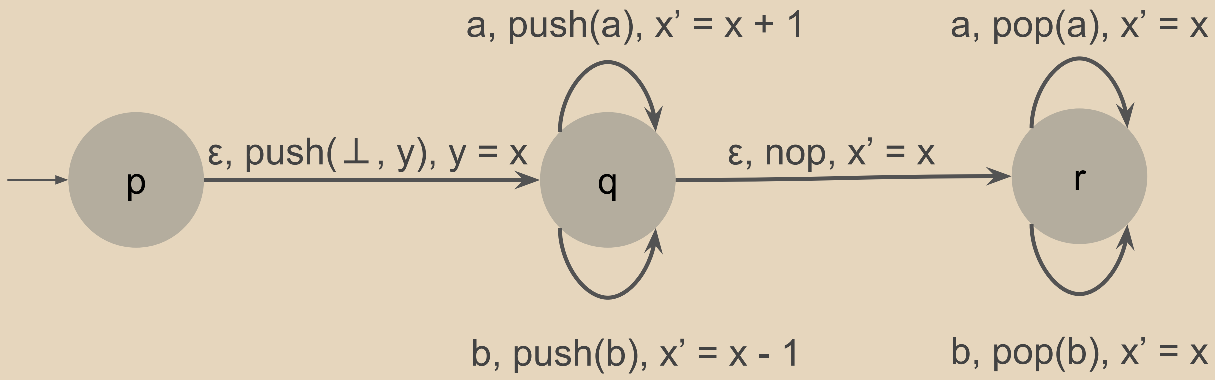
Untimed palindromes *with the same number of a's and b's*.



not **a context-free language**

Example (2)

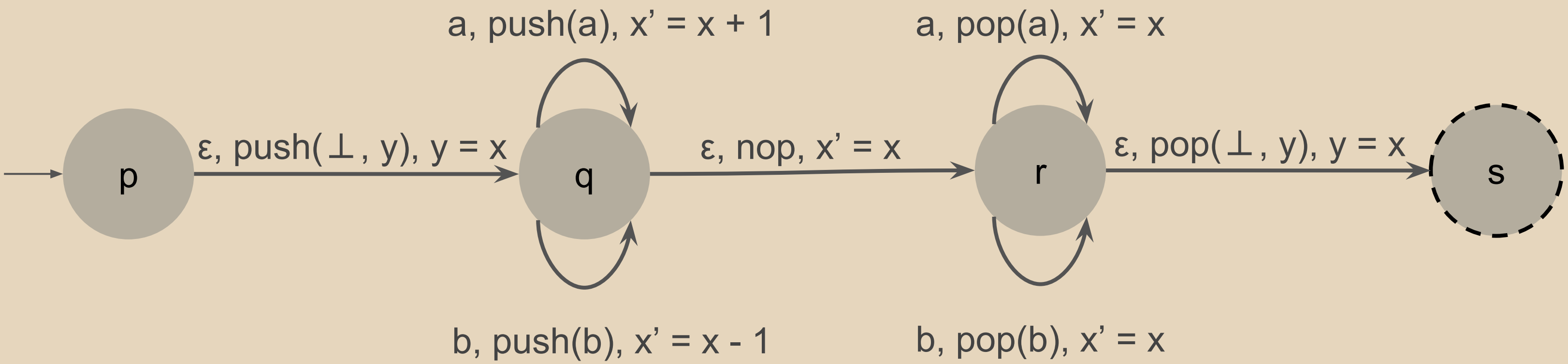
Untimed palindromes *with the same number of a's and b's*.



not a context-free language

Example (2)

Untimed palindromes *with the same number of a's and b's*.



***not* a context-free language**

Deciding reachability

Timed automata	PSPACE

Deciding reachability

Timed automata	PSPACE
Pushdown timed automata (untimed stack)	EXPTIME

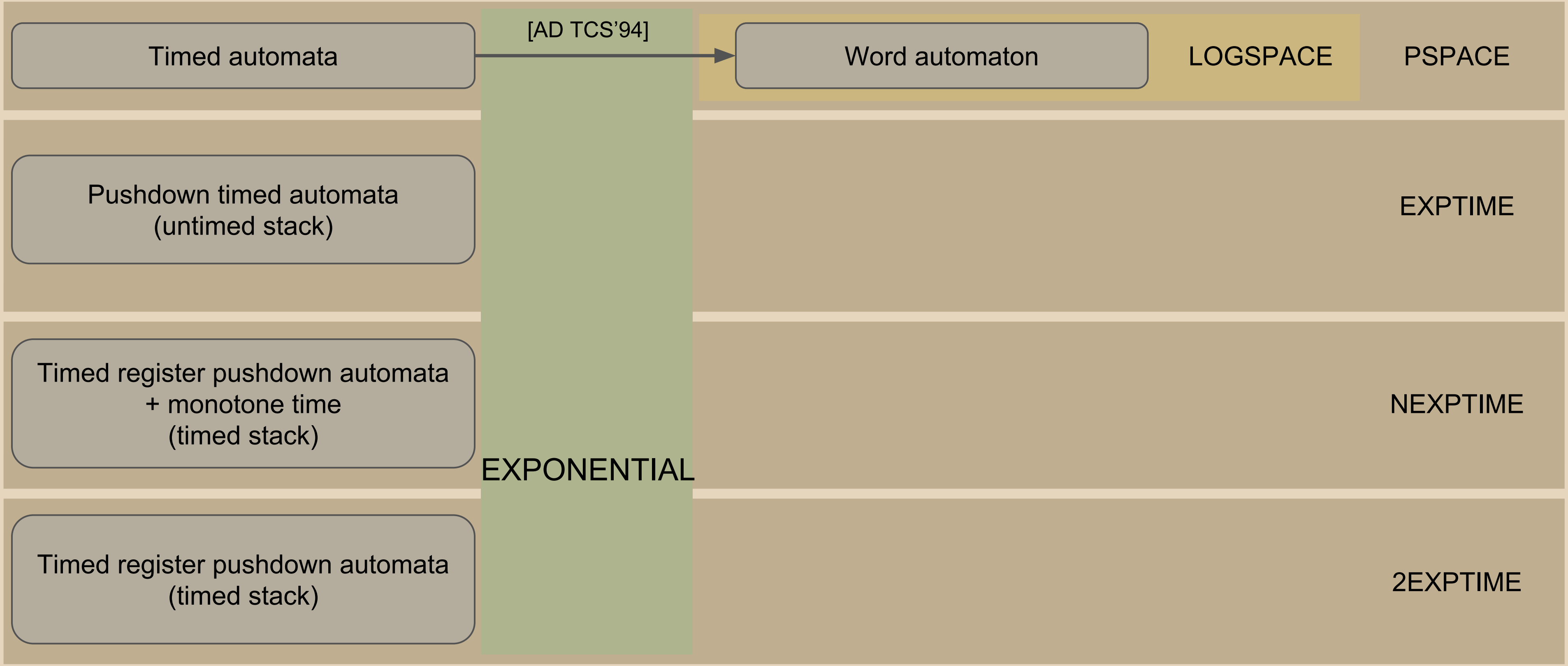
Deciding reachability

Timed automata	PSPACE
Pushdown timed automata (untimed stack)	EXPTIME
Timed register pushdown automata + monotone time (timed stack)	NEXPTIME

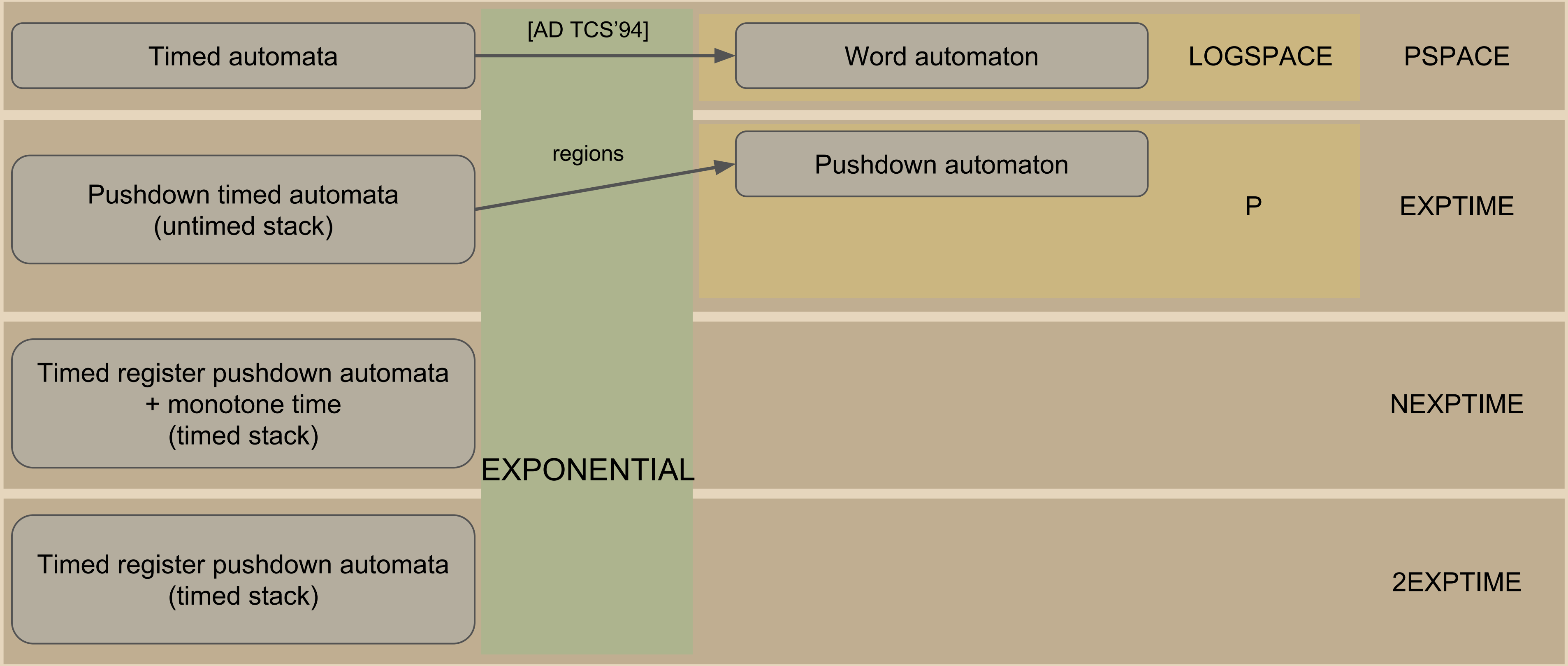
Deciding reachability

Timed automata	PSPACE
Pushdown timed automata (untimed stack)	EXPTIME
Timed register pushdown automata + monotone time (timed stack)	NEXPTIME
Timed register pushdown automata (timed stack)	2EXPTIME

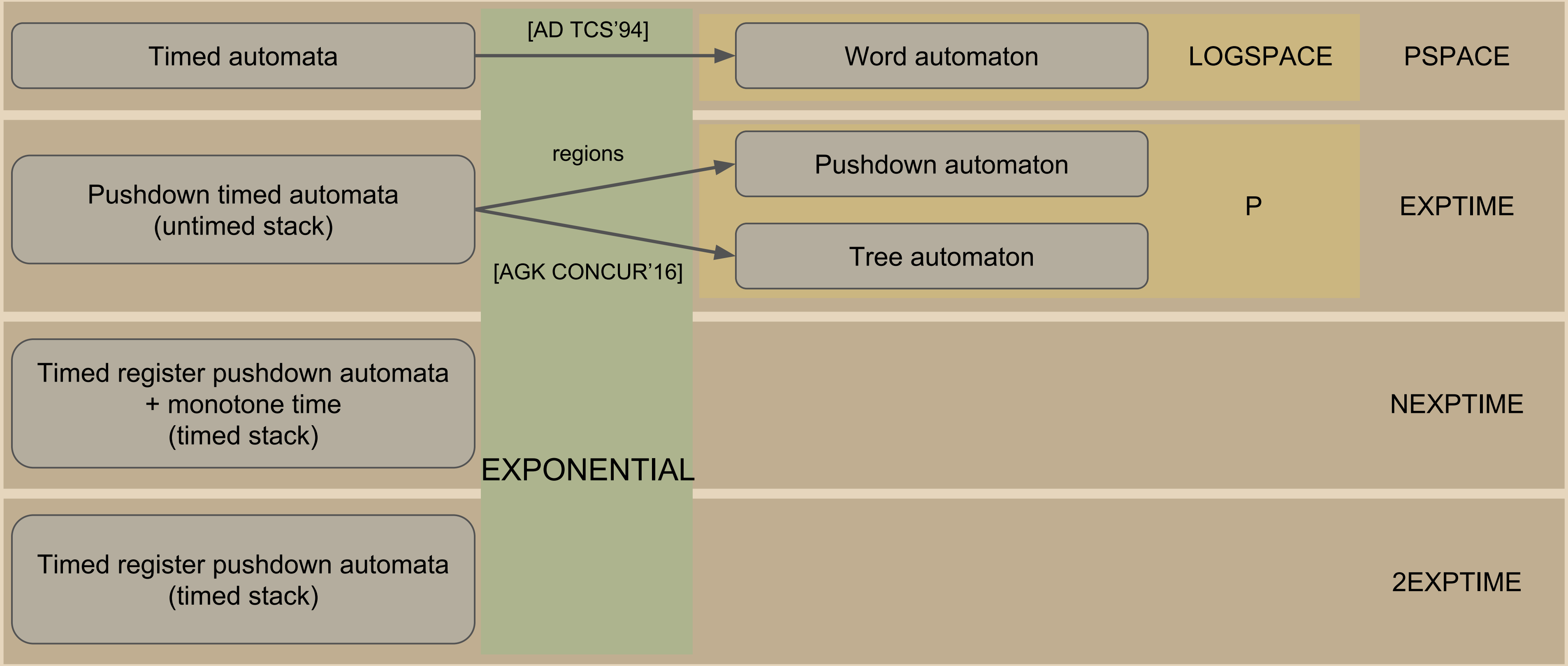
Deciding reachability



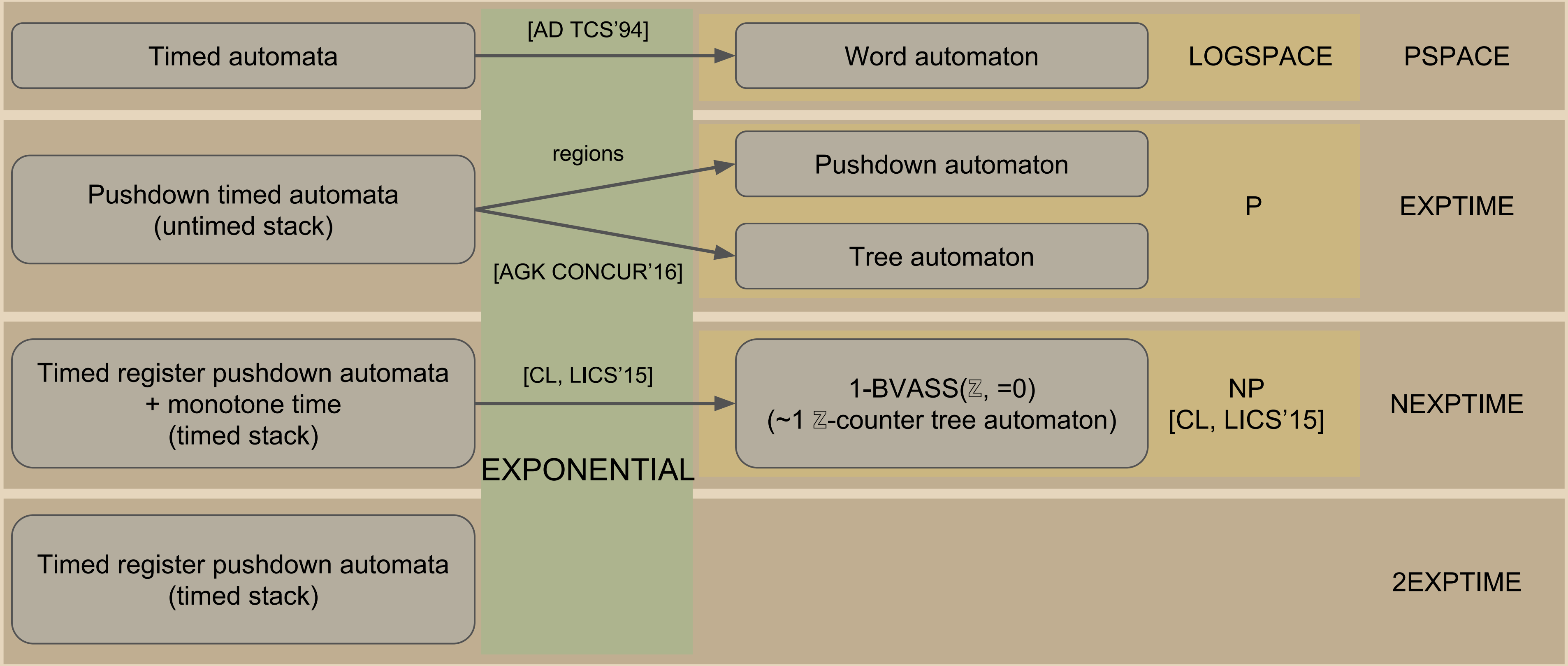
Deciding reachability



Deciding reachability



Deciding reachability



Deciding reachability

Timed automata	[AD TCS'94]	Word automaton	LOGSPACE	PSPACE
Pushdown timed automata (untimed stack)	regions [AGK CONCUR'16]	Pushdown automaton; Tree automaton	P	EXPTIME
Timed register pushdown automata + monotone time (timed stack)	[CL, LICS'15]	1-BVASS($\mathbb{Z}$, =0) (~1 $\mathbb{Z}$-counter tree automaton)	NP [CL, LICS'15]	NEXPTIME
	EXPONENTIAL			
Timed register pushdown automata (timed stack)	[CLLM, LICS'17]	1-BVASS($\mathbb{Z}$, ≥0, ≤0)	EXPTIME [CLLM, LICS'17]	2EXPTIME

Conclusions

To model time + recursion:

- Registers are seemingly more powerful than clocks.
- We get an expressive model with decidable non-emptiness (2EXPTIME).

Conclusions

To model time + recursion:

- Registers are seemingly more powerful than clocks.
- We get an expressive model with decidable non-emptiness (2EXPTIME).

Related models (not shown):

- Timed register context-free grammars (EXPTIME-c).

Conclusions

To model time + recursion:

- Registers are seemingly more powerful than clocks.
- We get an expressive model with decidable non-emptiness (2EXPTIME).

Related models (not shown):

- Timed register context-free grammars (EXPTIME-c).

Open questions:

- We have only an EXPTIME lower bound for our trPDA model.
- 1-BVASS($\mathbb{Z}$, ≥ 0, ≤ 0) are in EXPTIME and PSPACE-hard.
- Truly expressive timed pushdown automata *with clocks*?