

Automata for Real-Time Systems

B. Srivathsan

Chennai Mathematical Institute

Let $T\Sigma^*$ denote the set of **all timed words**

Universality: Given A , is $\mathcal{L}(A) = T\Sigma^*$?

Inclusion: Given A, B , is $\mathcal{L}(B) \subseteq \mathcal{L}(A)$?

Universality and inclusion are **undecidable** when A has **two clocks** or more

A theory of timed automata

Alur and Dill. *TCS*'94

A decidable case of the inclusion problem

Universality: Given A , is $\mathcal{L}(A) = T\Sigma^*$?

Inclusion: Given A, B , is $\mathcal{L}(B) \subseteq \mathcal{L}(A)$?

One-clock restriction

Universality and inclusion are **decidable** when A has at most **one clock**

On the language inclusion problem for timed automata: Closing a decidability gap

Ouaknine and Worrell. *LICS'05*

Universality: Given A , is $\mathcal{L}(A) = T\Sigma^*$?

Inclusion: Given A, B , is $\mathcal{L}(B) \subseteq \mathcal{L}(A)$?

One-clock restriction

Universality and inclusion are **decidable** when A has at most **one clock**

On the language inclusion problem for timed automata: Closing a decidability gap

Ouaknine and Worrell. *LICS*'05

In this lecture: **universality** for one clock TA

Step 0:

Well-quasi orders and Higman's Lemma

Quasi-order

Given a set Q , a **quasi-order** is a **reflexive** and **transitive** relation:

$$\sqsubseteq \subseteq Q \times Q$$

- ▶ $(\mathbb{N}, \leq)$
- ▶ $(\mathbb{Z}, \leq)$

Let $\Lambda = \{A, B, \dots, Z\}$, $\Lambda^* = \{\text{set of words}\}$

- ▶ $(\Lambda^*, \text{lexicographic order } \sqsubseteq_L)$: $AAAB \sqsubseteq_L AAB \sqsubseteq_L AB$
- ▶ $(\Lambda^*, \text{prefix order } \sqsubseteq_P)$: $AB \sqsubseteq_P ABA \sqsubseteq_P ABAA$
- ▶ $(\Lambda^*, \text{subword order } \preceq)$ $HIGMAN \preceq \text{HIGHMOUNTAIN}$ [OW'05]

Well-quasi-order

An infinite sequence $\langle q_1, q_2, \dots \rangle$ in $(\mathcal{Q}, \sqsubseteq)$ is **sat** if $\exists i < j : q_i \sqsubseteq q_j$

A quasi-order $\sqsubseteq$ is a **well-quasi-order (wqo)** if **every** infinite sequence is **sat**

- ▶ $(\mathbb{N}, \leq)$
- ▶ $(\mathbb{Z}, \leq)$
- ▶ $(\Lambda^*, \text{lexicographic order } \sqsubseteq_L)$:
- ▶ $(\Lambda^*, \text{prefix order } \sqsubseteq_P)$:
- ▶ $(\Lambda^*, \text{subword order } \preceq)$

Well-quasi-order

An infinite sequence $\langle q_1, q_2, \dots \rangle$ in $(\mathcal{Q}, \sqsubseteq)$ is **saturating** if $\exists i < j : q_i \sqsubseteq q_j$

A quasi-order $\sqsubseteq$ is a **well-quasi-order (wqo)** if **every** infinite sequence is saturating

- ▶ $(\mathbb{N}, \leq)$ ✓
- ▶ $(\mathbb{Z}, \leq)$ ✗ $-1 \geq -2 \geq -3, \dots$
- ▶ $(\Lambda^*, \text{lexicographic order } \sqsubseteq_L)$: ✗ $B \sqsubseteq_L AB \sqsubseteq_L AAB \dots$
- ▶ $(\Lambda^*, \text{prefix order } \sqsubseteq_P)$: ✗ $B, AB, AAB, \dots$
- ▶ $(\Lambda^*, \text{subword order } \preceq)$

Well-quasi-order

An infinite sequence $\langle q_1, q_2, \dots \rangle$ in $(\mathcal{Q}, \sqsubseteq)$ is **saturating** if $\exists i < j : q_i \sqsubseteq q_j$

A quasi-order $\sqsubseteq$ is a **well-quasi-order (wqo)** if **every** infinite sequence is saturating

- ▶ $(\mathbb{N}, \leq)$ ✓
- ▶ $(\mathbb{Z}, \leq)$ ✗ $-1 \geq -2 \geq -3, \dots$
- ▶ $(\Lambda^*, \text{lexicographic order } \sqsubseteq_L)$: ✗ $B \sqsubseteq_L AB \sqsubseteq_L AAB \dots$
- ▶ $(\Lambda^*, \text{prefix order } \sqsubseteq_P)$: ✗ $B, AB, AAB, \dots$
- ▶ $(\Lambda^*, \text{subword order } \preceq)$?

Higman's lemma

Let $\sqsubseteq$ be a quasi-order on Λ

Define the induced **monotone domination order** $\preceq$ on Λ^* as follows:

$$a_1 \dots a_m \preceq b_1 \dots b_n \quad \text{if there exists a **strictly increasing** function}$$
$$f : \{1, \dots, m\} \mapsto \{1, \dots, n\} \text{ s.t.}$$
$$\forall 1 \leq i \leq m : a_i \sqsubseteq b_{f(i)}$$

Higman's lemma

Let $\sqsubseteq$ be a quasi-order on Λ

Define the induced **monotone domination order** $\preceq$ on Λ^* as follows:

$$a_1 \dots a_m \preceq b_1 \dots b_n \quad \text{if there exists a strictly increasing function} \\ f : \{1, \dots, m\} \mapsto \{1, \dots, n\} \text{ s.t.} \\ \forall 1 \leq i \leq m : a_i \sqsubseteq b_{f(i)}$$

Higman'52

If $\sqsubseteq$ is a wqo on Λ , then the induced monotone domination order $\preceq$ is a wqo on Λ^*

Subword order

$$\Lambda := \{A, B, \dots, Z\}$$

$$\sqsubseteq := x \sqsubseteq y \text{ if } x = y$$

Subword order

$$\Lambda := \{A, B, \dots, Z\}$$

$$\sqsubseteq := x \sqsubseteq y \text{ if } x = y$$

$\sqsubseteq$ is a **wqo** as Λ is **finite**

Subword order

$$\Lambda := \{A, B, \dots, Z\}$$

$$\sqsubseteq := x \sqsubseteq y \text{ if } x = y$$

$\sqsubseteq$ is a **wqo** as Λ is **finite**

Induced monotone domination order $\preceq$ is the subword order

HIGMAN $\preceq$ *HIGHMOUNTAIN*

Subword order

$$\Lambda := \{A, B, \dots, Z\}$$

$$\sqsubseteq := x \sqsubseteq y \text{ if } x = y$$

$\sqsubseteq$ is a **wqo** as Λ is **finite**

Induced monotone domination order $\preceq$ is the subword order

$$HIGMAN \preceq HIGHMOUNTAIN$$

By Higman's lemma, $\preceq$ is a wqo too

If we start writing an **infinite sequence** of words, we will **eventually** write down a **superword** of an earlier word in the sequence

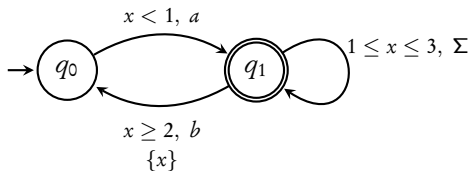
Step 1:

**A naive procedure for universality of one-clock
TA**

Terminology

Let $A = (Q, \Sigma, Q_0, \{x\}, T, F)$ be a timed automaton with one clock

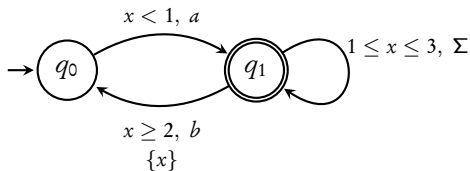
- ▶ **Location:** $q_0, q_1, \dots \in Q$
- ▶ **State:** (q, u) where $u \in \mathbb{R}_{\geq 0}$ gives value of the clock
- ▶ **Configuration:** **finite** set of states



Terminology

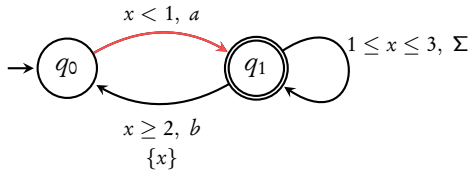
Let $A = (Q, \Sigma, Q_0, \{x\}, T, F)$ be a timed automaton with one clock

- **Location:** $q_0, q_1, \dots \in Q$
- **State:** (q, u) where $u \in \mathbb{R}_{\geq 0}$ gives value of the clock
- **Configuration:** **finite** set of states $\{(q_1, 2.3), (q_0, 0)\}$



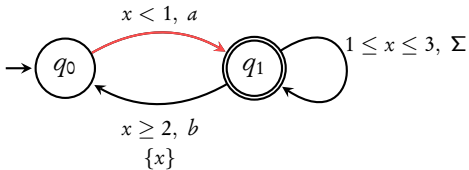
Transition between configurations:

$$\{(q_0, 0)\} \xrightarrow{0.2, a}$$



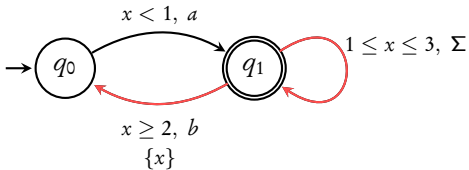
Transition between configurations:

$$\{(q_0, 0)\} \xrightarrow{0.2, a} \{(q_1, 0.2)\}$$



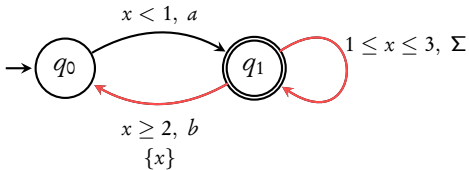
Transition between configurations:

$$\{(q_0, 0)\} \xrightarrow{0.2, a} \{(q_1, 0.2)\} \xrightarrow{2.1, b}$$



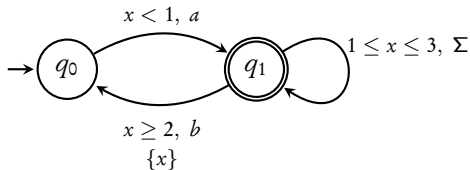
Transition between configurations:

$$\{(q_0, 0)\} \xrightarrow{0.2, a} \{(q_1, 0.2)\} \xrightarrow{2.1, b} \{(q_1, 2.3), (q_0, 0)\} \dots$$



Transition between configurations:

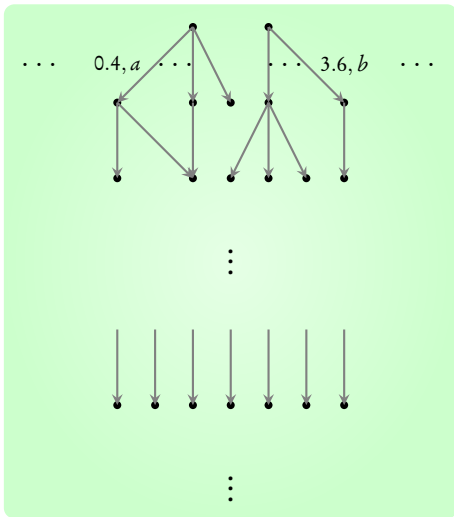
$$\{(q_0, 0)\} \xrightarrow{0.2, a} \{(q_1, 0.2)\} \xrightarrow{2.1, b} \{(q_1, 2.3), (q_0, 0)\} \dots$$



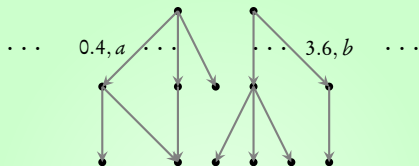
$$C_1 \xrightarrow{\delta, a} C_2 \text{ if}$$

$$C_2 = \{ (q_2, u_2) \mid \exists (q_1, u_1) \in C_1 \text{ s. t. } (q_1, u_1) \xrightarrow{\delta, a} (q_2, u_2) \}$$

Labeled transition system of configurations



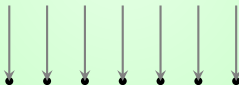
Labeled transition system of configurations



$\vdots$

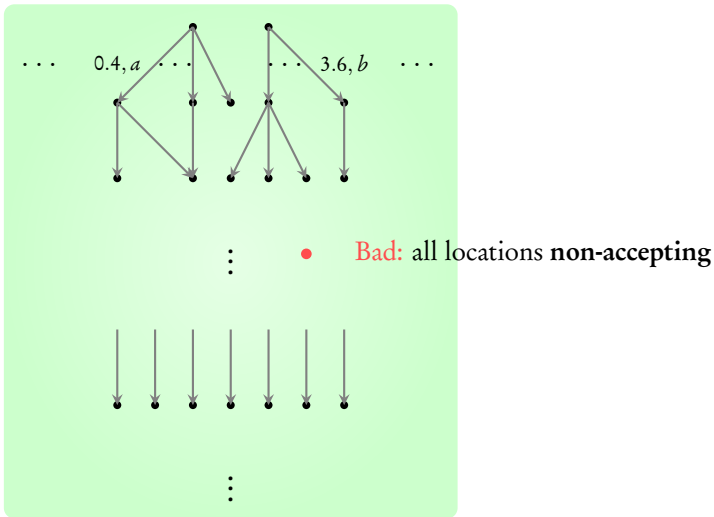


Bad: all locations **non-accepting**

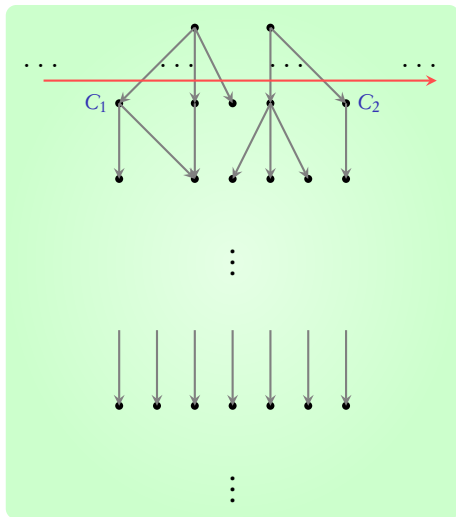


$\vdots$

Labeled transition system of configurations



Is a **bad** configuration **reachable** from some **initial** configuration?

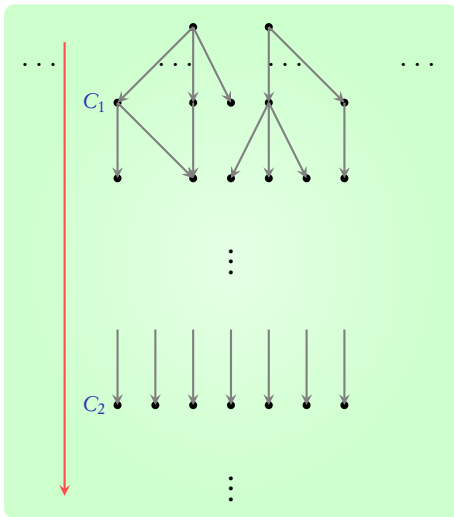


abstraction by equivalence $\sim$

$C_1 \sim C_2$ iff:

C_1 goes to a **bad** config. $\Leftrightarrow C_2$ goes to a **bad** config.

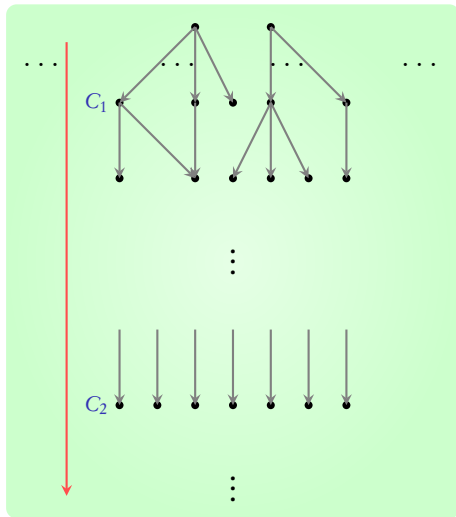
finite **domination** order $\succsim$



$C_1 \succsim C_2$ iff:

C_2 goes to a **bad** config $\Rightarrow$ C_1 goes to a **bad** config. too

finite **domination** order $\preccurlyeq$



$C_1 \preccurlyeq C_2$ iff:

C_2 goes to a **bad** config $\Rightarrow$ C_1 goes to a **bad** config. too

No need to explore C_2 !

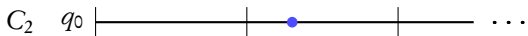
Step 2:

The equivalence

Credits: Examples in this part taken from one of **Ouaknine's** talks

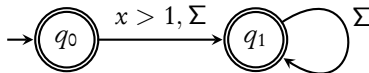
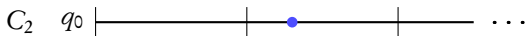
Equivalent configurations: Examples

$$C_1 = \{(q_0, 0.5)\} \approx C_2 = \{(q_0, 1.3)\}$$

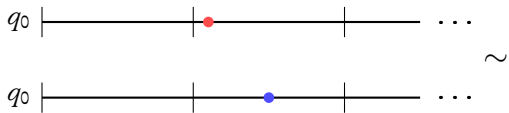
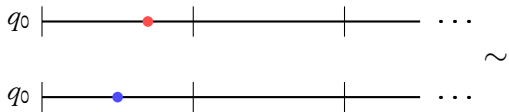


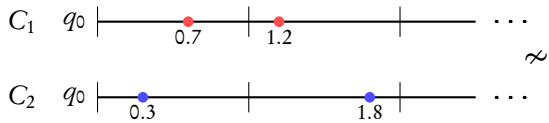
Equivalent configurations: Examples

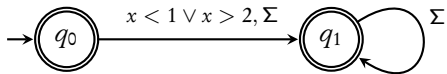
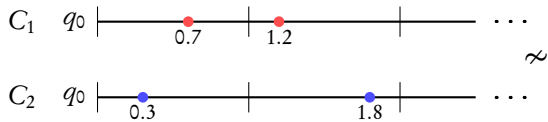
$$C_1 = \{(q_0, 0.5)\} \approx C_2 = \{(q_0, 1.3)\}$$



C_2 is universal, but C_1 **rejects** $(a, 0)$



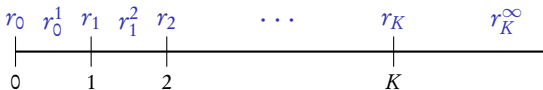




C_2 is universal, but C_1 **rejects** $(a, 0.5)$

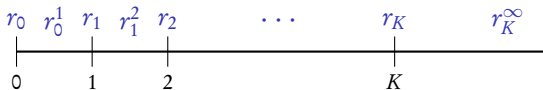
Let K be the largest constant appearing in A

Define $REG = \{r_0, r_0^1, r_1, \dots, r_K, r_K^\infty\}$



Let K be the largest constant appearing in A

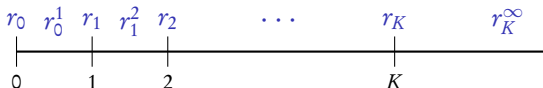
Define $REG = \{r_0, r_0^1, r_1, \dots, r_K, r_K^\infty\}$



$$C = \{(q_1, 0.0), (q_1, 0.3), (q_1, 1.2), (q_2, 1.0), (q_3, 0.8), (q_3, 1.3)\}$$

Let K be the largest constant appearing in A

Define $REG = \{r_0, r_0^1, r_1, \dots, r_K, r_K^\infty\}$

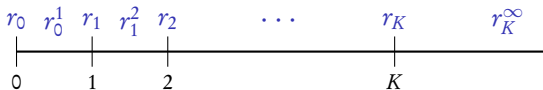


$$C = \{(q_1, 0.0), (q_1, 0.3), (q_1, 1.2), (q_2, 1.0), (q_3, 0.8), (q_3, 1.3)\}$$

$$\{(q_1, r_0, 0), (q_1, r_0^1, 0.3), (q_1, r_1^2, 0.2), (q_2, r_1, 0), (q_3, r_0^1, 0.8), (q_3, r_1^2, 0.3)\}$$

Let K be the largest constant appearing in A

Define $REG = \{r_0, r_0^1, r_1, \dots, r_K, r_K^\infty\}$



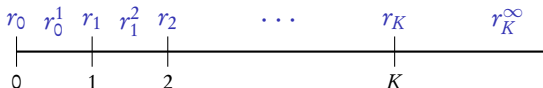
$$C = \{(q_1, 0.0), (q_1, 0.3), (q_1, 1.2), (q_2, 1.0), (q_3, 0.8), (q_3, 1.3)\}$$

$$\{(q_1, r_0, 0), (q_1, r_0^1, 0.3), (q_1, r_1^2, 0.2), (q_2, r_1, 0), (q_3, r_0^1, 0.8), (q_3, r_1^2, 0.3)\}$$

$$\{(q_1, r_0, 0), (q_2, r_1, 0)\} \{(q_1, r_1^2, 0.2)\} \{(q_1, r_0^1, 0.3)(q_3, r_1^2, 0.3)\} \{(q_3, r_0^1, 0.8)\}$$

Let K be the largest constant appearing in A

Define $REG = \{r_0, r_0^1, r_1, \dots, r_K, r_K^\infty\}$



$$C = \{(q_1, 0.0), (q_1, 0.3), (q_1, 1.2), (q_2, 1.0), (q_3, 0.8), (q_3, 1.3)\}$$

$$\{(q_1, r_0, 0), (q_1, r_0^1, 0.3), (q_1, r_1^2, 0.2), (q_2, r_1, 0), (q_3, r_0^1, 0.8), (q_3, r_1^2, 0.3)\}$$

$$\{(q_1, r_0, 0), (q_2, r_1, 0)\} \{(q_1, r_1^2, 0.2)\} \{(q_1, r_0^1, 0.3)(q_3, r_1^2, 0.3)\} \{(q_3, r_0^1, 0.8)\}$$

$$H(C) = \{(q_1, r_0), (q_2, r_1)\} \{(q_1, r_1^2)\} \{(q_1, r_0^1)(q_3, r_1^2)\} \{(q_3, r_0^1)\}$$

Let K be the largest constant appearing in A

$$REG := \{r_0, r_0^1, r_1, \dots, r_K, r_K^\infty\}$$

$$\Lambda := \mathcal{P}(Q \times REG)$$

We can give $H : C \mapsto \Lambda^*$ that remembers:

- ▶ **integral** part of the clock value (modulo K) in each state of C ,
- ▶ **order of fractional** parts of the clock among different states in C

Equivalence

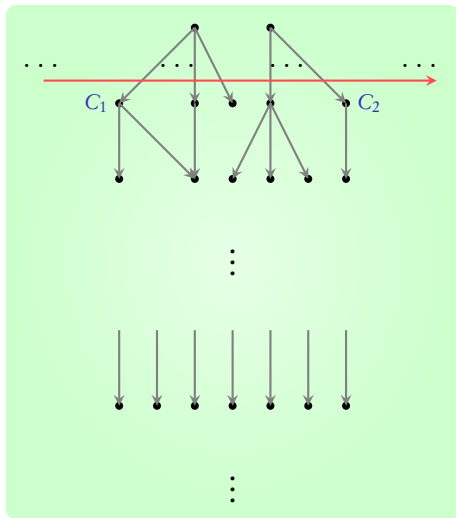
$$C_1 \sim C_2 \text{ if } H(C_1) = H(C_2)$$

Equivalence

$$C_1 \sim C_2 \text{ if } H(C_1) = H(C_2)$$

It can be shown that $\sim$ is a **bisimulation**

C_1 goes to a **bad** config. $\Leftrightarrow C_2$ goes to a **bad** config.



abstraction by equivalence $\sim$

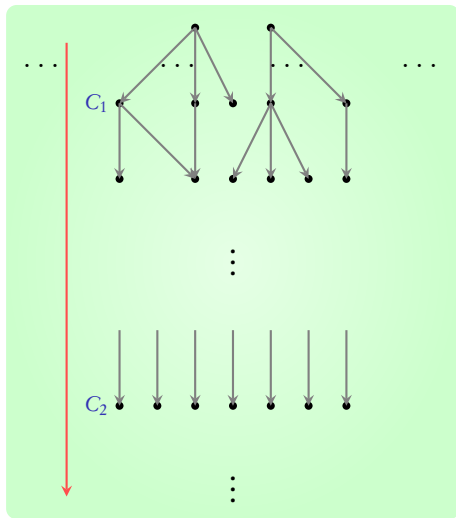
$C_1 \sim C_2$ iff:

C_1 goes to a **bad** config. $\Leftrightarrow C_2$ goes to a **bad** config.

Step 3:

The domination order

finite **domination** order $\preceq$



$C_1 \preceq C_2$ iff:

C_2 goes to a **bad** config $\Rightarrow$ C_1 goes to a **bad** config. too

Look at $H(C_1)$ and $H(C_2)$, the words over Λ^*

$$\Lambda = \mathcal{P}(Q \times REG)$$

Look at $H(C_1)$ and $H(C_2)$, the words over Λ^*

$$\Lambda = \mathcal{P}(Q \times REG)$$

Let $\subseteq$ be the **inclusion** (quasi-)order on Λ

Look at $H(C_1)$ and $H(C_2)$, the words over Λ^*

$$\Lambda = \mathcal{P}(Q \times REG)$$

Let $\subseteq$ be the **inclusion** (quasi-)order on Λ

Consider the induced monotone domination order $\preceq$ over Λ^*

$$\begin{aligned} & \{(q_0, r_0)\} \subseteq \{(q_1, r_0^1), (q_0, r_2^3)\} \\ & \preceq \\ & \{(q_0, r_0), (q_1, r_1)\} \subseteq \{(q_2, r_2^3)\} \subseteq \{(q_1, r_0^1), (q_0, r_2^3), (q_2, r_1^2)\} \end{aligned}$$

Look at $H(C_1)$ and $H(C_2)$, the words over Λ^*

$$\Lambda = \mathcal{P}(Q \times REG)$$

Let $\subseteq$ be the **inclusion** (quasi-)order on Λ

Consider the induced monotone domination order $\preceq$ over Λ^*

$$\{(q_0, r_0)\} \subseteq \{(q_1, r_0^1), (q_0, r_2^3)\}$$

$$\preceq$$

$$\{(q_0, r_0), (q_1, r_1)\} \subseteq \{(q_2, r_2^3)\} \subseteq \{(q_1, r_0^1), (q_0, r_2^3), (q_2, r_1^2)\}$$

Theorem: If $H(C_1) \preceq H(C_2)$, then $\exists C'_2 \subseteq C_2$ s.t. $C_1 \sim C'_2$

Look at $H(C_1)$ and $H(C_2)$, the words over Λ^*

$$\Lambda = \mathcal{P}(Q \times REG)$$

Let $\subseteq$ be the **inclusion** (quasi-)order on Λ

Consider the induced monotone domination order $\preceq$ over Λ^*

$$\{(q_0, r_0)\} \subseteq \{(q_1, r_0^1), (q_0, r_2^3)\}$$
$$\preceq$$

$$\{(q_0, r_0), (q_1, r_1)\} \subseteq \{(q_2, r_2^3)\} \subseteq \{(q_1, r_0^1), (q_0, r_2^3), (q_2, r_1^2)\}$$

Theorem: If $H(C_1) \preceq H(C_2)$, then $\exists C'_2 \subseteq C_2$ s.t. $C_1 \sim C'_2$

$\subseteq$ is a wqo as Λ is **finite**. Therefore, $\preceq$ is a **wqo** due to **Higman's lemma**

Final algorithm

- ▶ Start from $H(C_0)$, where C_0 is the **initial configuration**
- ▶ Successor computation is **effective**
- ▶ Termination guaranteed as **domination order is wqo**

A is **universal** iff the algorithm does **not reach a bad node**

One-clock

Universality is **decidable** for one-clock timed automata

One-clock

Universality is **decidable** for one-clock timed automata

For **two clocks**, we know universality is undecidable

One-clock

Universality is **decidable** for one-clock timed automata

For **two clocks**, we know universality is undecidable

Where does this algorithm go wrong when A has two clocks?

Two clocks

State: (q, u, v)

Configuration: $\{(q_1, u_1, v_1), (q_2, u_2, v_2), \dots, (q_n, u_n, v_n)\}$

At the **least**, the following should be remembered while abstracting:

- ▶ relative ordering between fractional parts of x
- ▶ relative ordering between fractional parts of y

Current encoding can remember **only one** of them

Other encodings possible?

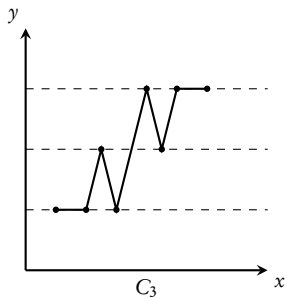
Consider some domination order $\preceq$

$C_1 \not\preceq C_2$ if for all $C'_2 \subseteq C_2$:

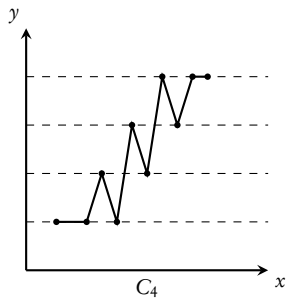
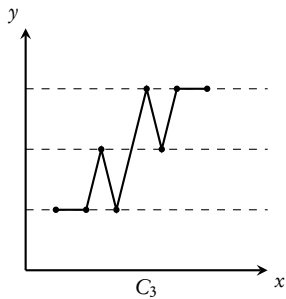
- ▶ either relative order of clock x does not match
- ▶ or relative order of clock y does not match

In the next slide: **No wqo possible!**

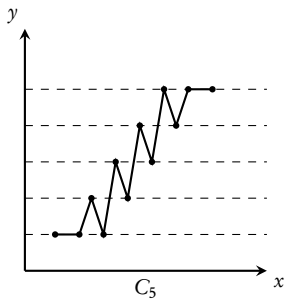
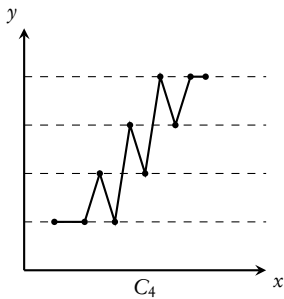
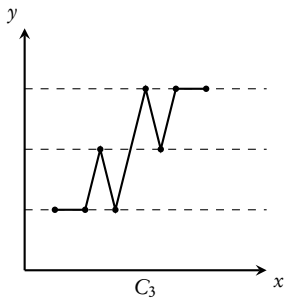
An infinite **non-saturating** sequence $C_1, C_2, C_3, \dots$



An infinite **non-saturating** sequence $C_1, C_2, C_3, \dots$



An infinite **non-saturating** sequence $C_1, C_2, C_3, \dots$



Conclusion

- ▶ An algorithm for **universality** when A has one clock
- ▶ Can be **extended** for $\mathcal{L}(B) \subseteq \mathcal{L}(A)$ when A has one-clock