TIMED AUTOMATA

LECTURE 17

Recall:

Configuration C — Word $H(C)$

$M = 10$

$\{(q_1, 2.7), (q_2, 4.5)\ (q_1, 7.5)\}$ — $\{(q_2, r_4^5)\ (q_1, r_7^8)\}\ \{(q_1, r_2^3)\}$

$\{(q_0, 10.5), (q_1, 7.7), (q_3, 4.2), (q_4, 10)\}$ $(q_5, 5)$ — $\{(q_4, r_{10})\ (q_5, r_5)\}\ \{(q_3, r_4^5)\}$ $\{(q_1, r_7^8)\}\ \{(q_0, r_{10}^{\infty})\}$

Equivalence: Assume a fixed M.

$C_1 \sim_H C_2$ if $H(C_1) = H(C_2)$

Lemma 1: Let $C_1 \sim_H C_2$. Then for every time elapse $\delta \geq 0$, there exists $\delta' \geq 0$ s.t.

$C_1 + \delta \sim_H C_2 + \delta'$

Lemma 2: Let $C_1 \sim_H C_2$. If $C_1 \xrightarrow{a} C_1'$ then $C_2 \xrightarrow{a} C_2'$ s.t. $C_1' \sim_H C_2'$

Corollary: Let $C_1 \sim_H C_2$. If $C_1 \xrightarrow{\delta, a} C_1'$ then there exists δ': $C_2 \xrightarrow{\delta', a} C_2'$ s.t. $C_1' \sim_H C_2'$

Next: Constructing a graph whose nodes are words of the form $H(C)$.

Graph over H(c):

Initial configuration C_0: $\{ (q_0^1, 0), (q_0^2, 0), \ldots (q_0^k, 0) \}$

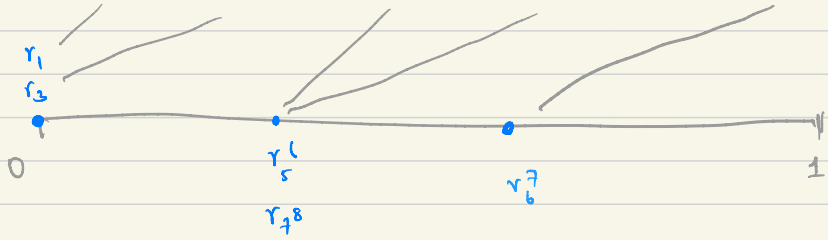
$$H(C_0) = \{ (q_0^1, r_0) (q_0^2, r_0) \ldots (q_0^k, r_0) \}$$

Successors:

i) Time successors. $(M = 10)$

Ex: 1:

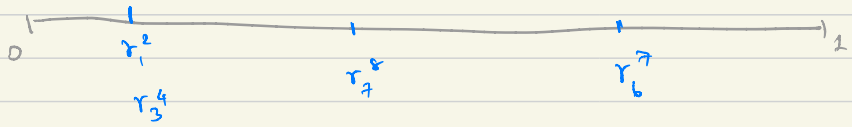
$\{ (q_0, r_1) (q_1, r_3) \} \{ (q_0, r_5^6) (q_2, r_7^8) \} \{ (q_4, r_6^7) \} \{ (q_5, r_{10}^\infty) \}$

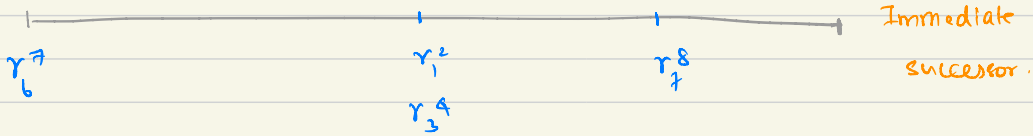


$\{ (q_0, r_1^2) (q_1, r_3^4) \} \{ (q_0, r_5^6) (q \quad r_7^8) \} \{ (q_4, r_6^7) \} \{ (q_5, r_{10}^\infty) \}$

Ex. 2:

$\{ (q_0, r_1^2) (q_1, r_3^4) \} \{ (q_2, r_7^8) \} \{ (q_3, r_6^7) \}$





$\{ (q_3, r_7) \} \{ (q_0, r_1^2) (q_1, r_3^4) \} \{ (q_2, r_7^8) \}$

Ex. 3: $M=10$

$\{(q_1, r_{10})\}$ $\{(q_2, r_1^2)\}$ $\{(q_3, r_3^4)(q_1, r_1^2)\}$

$\downarrow$

$\{(q_2, r_1^2)\}$ $\{(q_3, r_3^4)(q_1, r_1^2)\}$ $\{(q_1, r_{10}^{\infty})\}$

Remark:

Starting from $H(c)$ all its time-successors can be computed

$$H(c) \xrightarrow{\tau} H(c')$$

if $H(c')$ is a time-successor of $H(c)$.

Time-successor of $H(c)$:

c' is a time successor of c if $\exists \delta: c \xrightarrow{\delta} c'$

Time-successors of $H(c) = \{ H(c') \mid c' \text{ is a time-succ. of } c \}$

finite, and can be computed.

---

ii) Action successors:

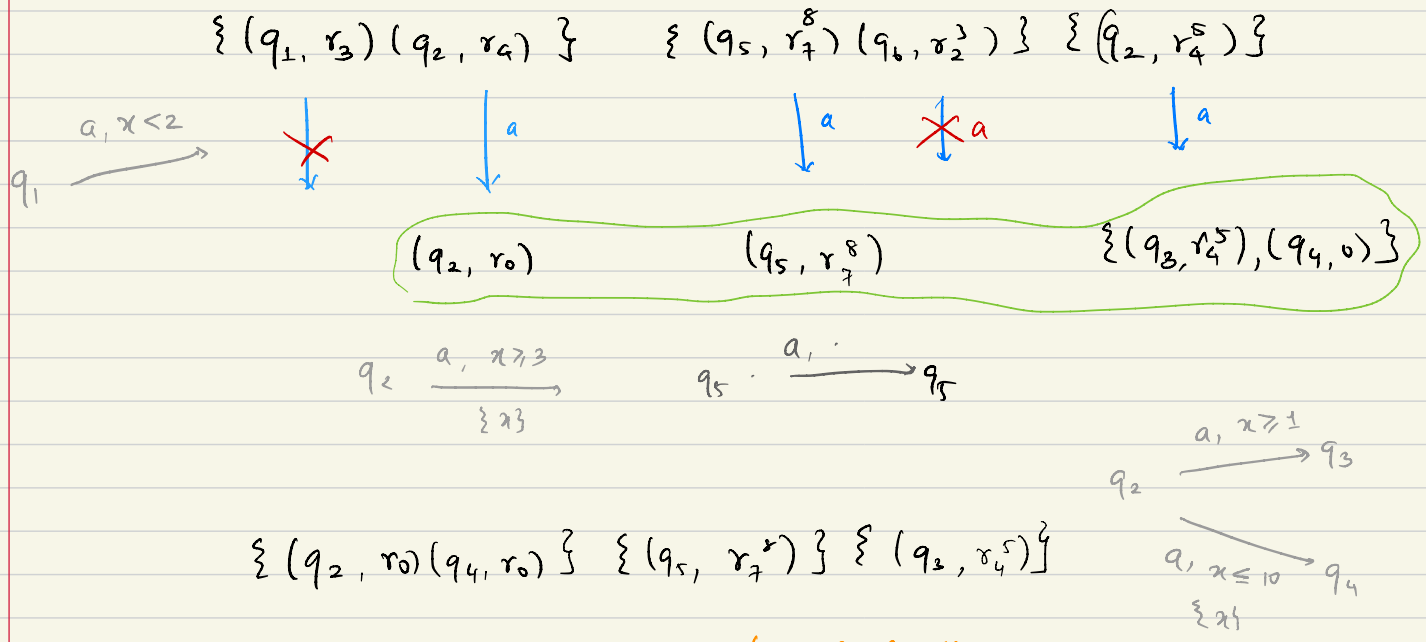
$$H(c) \xrightarrow{a} H(c') \quad \text{if}$$

$c \xrightarrow{a} c'$ is defined $c: \{(q_1, 5)(q_2, 4.7)(q_3, 7.1)\}$

a a a a ✗

$c' = \{(q_1, 0), (q_2, 4.7)\}$

Computing action-successors:



↳ a-successor.

Remark: Consider a word H. H = H(c1) = H(c2) ··· for several configurations.

Pick some c_i s.t. $H = H(c_i)$

- $c_i \xrightarrow{a} c_i'$

Then take $H(c_i')$ as the a-successor of H.

$$H \xrightarrow{a} H(c_i')$$

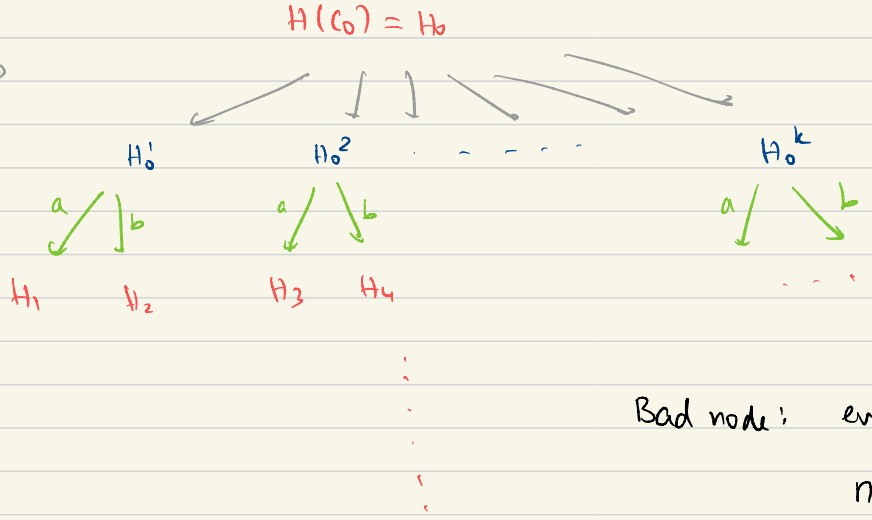
- The choice of c_i does not matter (Exercise)

Successors in the graph:

$H \xrightarrow{\tau} H'$ if H' is a time successor

$H \xrightarrow{a} H'$ if H' is an a-succ. of H.

$H(A)$ →



Bad node: every location is non-acc.

(Soundness)

Lemma: For every path:

$H_0 \xrightarrow{\tau, a_1} H_1 \xrightarrow{\tau, a_2} H_2 \longrightarrow \cdots \xrightarrow{\tau, a_n} H_n$

there exists a run over config:

$C_0 \xrightarrow{\delta_1, a_1} C_1 \xrightarrow{\delta_2, a_2} \cdots \xrightarrow{\delta_n, a_n} C_n$

s.t. $H(C_i) = H_i \quad \forall i \in \{1, \ldots, n\}$

(Completeness)

Lemma: For every run over config:

$C_0 \xrightarrow{\delta_1, a_1} C_1 \xrightarrow{\delta_2, a_2} \cdots \xrightarrow{\delta_n, a_n} C_n$

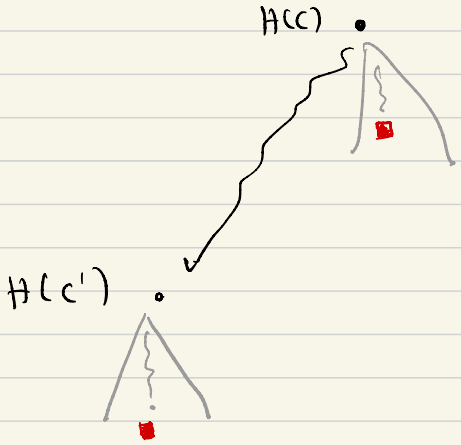
there exists a path:

$H_1 \xrightarrow{\tau, a_1} H_2 \xrightarrow{\tau, a_2} \cdots \xrightarrow{\tau, a_n} H_n$

s.t. $H(C_i) = H_i$

Proposition: A is not universal iff a bad node is reachable in $H(A)$

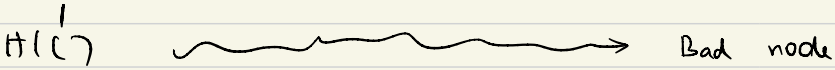
Termination

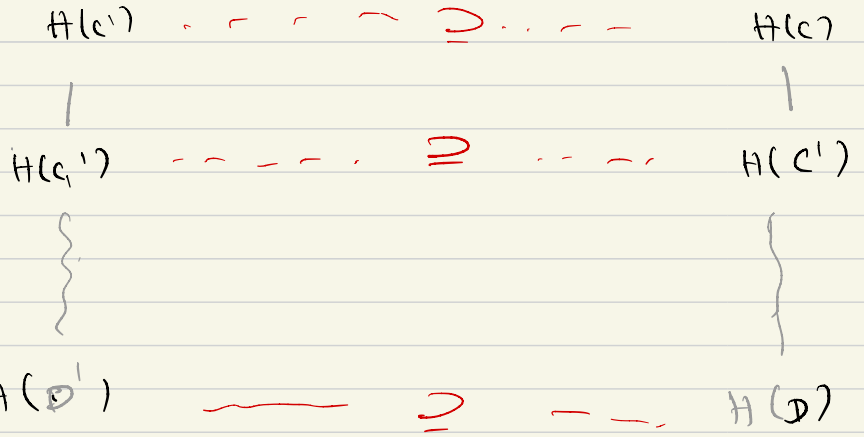


Ex $H(C) = \{(q_1, r_4^5)\}\ \{(q_2, r_7^8)\ (q_3, r_4^5)\}\ \{(q_1, r_{10}^{\infty})\}$

$H(C') = \{(q_2, r_4)\}\ \{(q_1, r_4^5)\ (q_7, r_3^4)\}\ \{(q_2, r_7^8), (q_3, r_4^5)\}$

$\{(q_1, r_{10}^{\infty}), (q_3, r_{10}^{\infty})\}$





Bad

Locations of H(D) are contained in location of H(D')

$\therefore$ If $H(D')$ is bad, then so is $H(D)$

The simulation relation:

$$H(c') \preceq H(c) :$$

$$H(c') = w_1' w_2' \cdots w_k' \preceq H(c) = w_1 w_2 \cdots w_n$$

if

there exists a strictly increasing function:

$$f: [1, \ldots n] \longmapsto [1, \ldots, k]$$

s.t.

$$w_i \subseteq w'_{f(i)}$$

$H(c) = \{(q_1, r_4^5)\} \{(q_2, r_7^8)(q_3, r_4^5)\} \{(q_1, r_{10}^{\infty})\}$

$H(c') = \{(q_2, r_4)\} \{(q_1, r_4^5)(q_7, r_3^4)\} \{(q_2, r_7^8), (q_3, r_4^5)\}$

$\{(q_1, r_{10}^{\infty}), (q_3, r_{10}^{\omega})\}$

Suppose Λ = Power-set $(Q \times REG)$ → finite ..

Then $H(c)$ is a word over Λ

- Consider the inclusion order in Λ.
- Then the simulation $\preceq$ is the induced monotone domination ordering, lifted from $\subseteq$.

Theorem: $\preceq$ is a wqo

(Higman's lemma + $\subseteq$ is a wqo over Λ)

Result:

Modify the enumeration:

if $H(c') \trianglelefteq H(c)$ for some earlier $H(c)$, ignore $H(c')$

else explore $H(c')$ further.

Since $\trianglelefteq$ is a wqo, there can be no infinite paths. The algorithm terminates.

→ This gives an algorithm for checking universality of one-clock TA.

Complexity? → left open in OW'05 paper.

- later proved to be non-primitive recursive by Lasota, Walukiewicz '08

(in a later class)

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

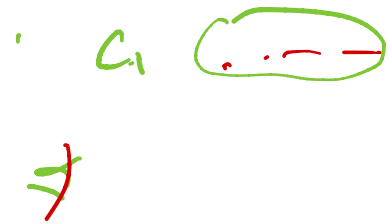
At the **least**, the following should be remembered while abstracting:

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

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

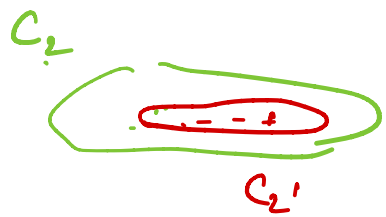
Other encodings possible?

Consider some domination order $\preccurlyeq$



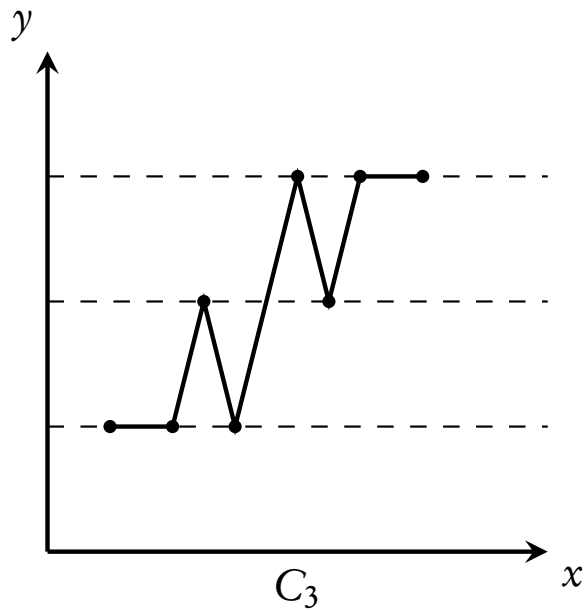
$C_1 \not\preccurlyeq C_2$ if for all $C_2' \subseteq C_2$:

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

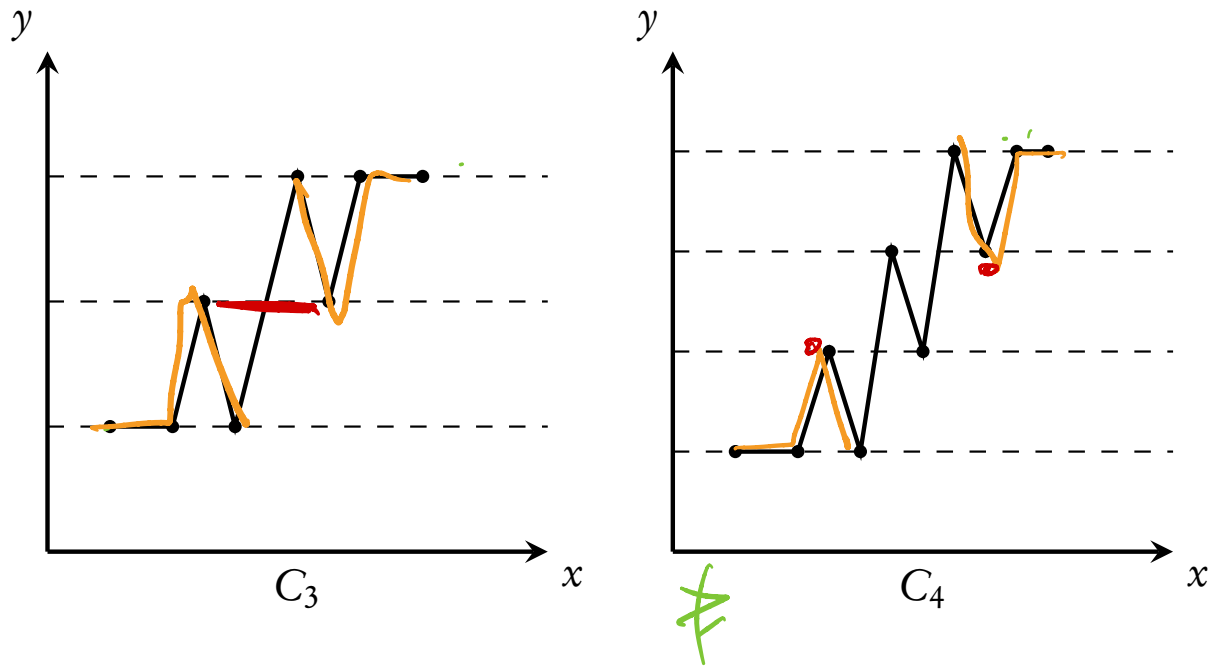


In the next slide: **No wqo** possible!

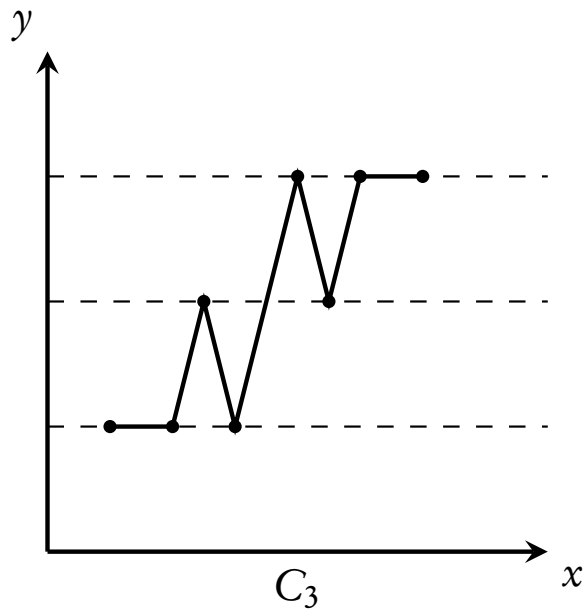
An infinite **non-saturating** sequence $C_1, C_2, C_3, \ldots$

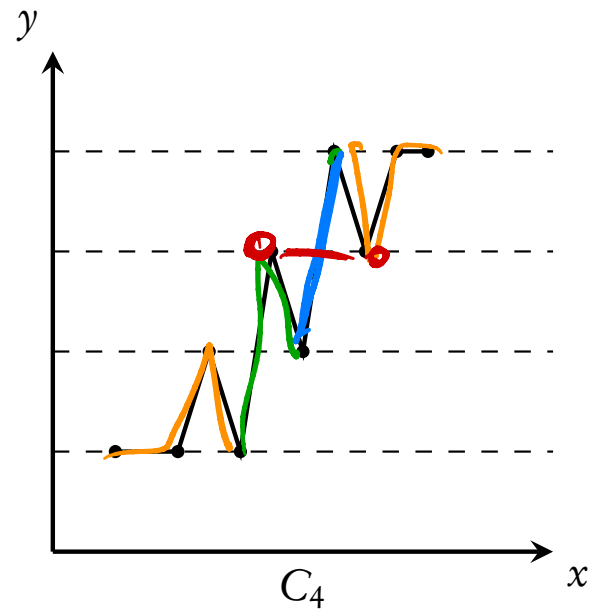


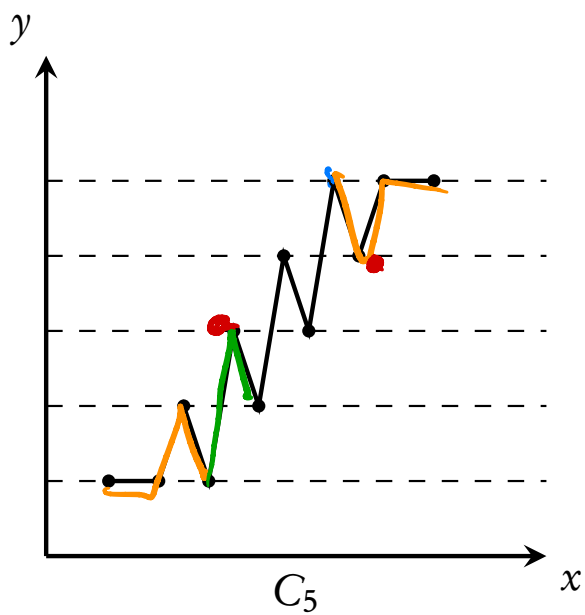
An infinite **non-saturating** sequence $C_1, C_2, C_3, \ldots$



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







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