TIMED AUTOMATA

LECTURE 15

Timed Automata with diagonal constraints:

Guards: $\circ \xrightarrow{x \leq c} \bullet$

$\varphi : \quad x \sim c \mid \varphi \wedge \varphi$ $\qquad x \in X$

$c \in \mathbb{N}$

$\sim \in \{<, \leq, >, \geq\}$

We will now consider "diagonal constraints" in guards:

$\circ \xrightarrow{x - y \leq 3 \,\wedge\, y \geq 5}$

$c \in \mathbb{N}$

$\varphi: \quad x \sim c \mid x - y \sim c \mid \varphi \wedge \varphi$ $\qquad \sim \in \{<, \leq, >, \geq\}$

$x, y \in X$

Questions:

- 1. Do diagonal constraints add expressive power?

  ~ No

- 2. Do diagonal constraints give succinctness?

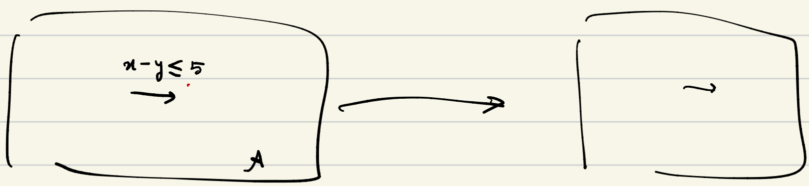
  – Yes.

Eliminating diagonal constraints:

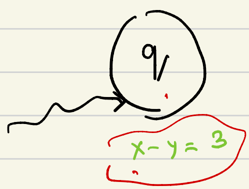
Terminology: Let us call timed automata with diagonal constraints as "d- timed automata"

Theorem: For every d- timed automaton $\mathcal{A}$, there exists a timed automaton (without diagonal constraints) $\mathcal{B}$ s.t.

$$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$$



Idea: To eliminate $\boxed{x-y \leq 5}$



Difference between two clocks stays constant during time elapse

At state 'q' either we reach with a value s.t. $x-y \leq 5$ is true.

or $x-y > 5$ (the constraint is false)

The idea is to split 'q' into two:

$(q, 1)$ and $(q, 0)$

$x - y \leq 5$ is true

$x - y \leq 5$ is false

$(x - y > 5)$

$$q \xrightarrow[\{x\}]{a} q'$$

- $(q, 1) \xrightarrow[\{x\}]{a} (q', 1)$

$x - y \leq 5 \dashrightarrow \boxed{0 - y \leq 5}$

- $(q, 0) \xrightarrow[\{x\}]{a} (q', 1)$

$x = 0$

$y \geq 0$ $\quad x - y = 0 - y \leq 5$

- $q \xrightarrow[\{y\}]{a} q'$

$(q, 1) \xrightarrow[\{y\}]{a,\ x \leq 5} (q', 1)$

$x - 0$

$(q, 1) \xrightarrow[\{y\}]{a,\ x > 5} (q', 0)$

- 1. Diagonal values do not change with time elapse

- 2. Diagonal values change during resets.

- 3. Maintain the truth of the diagonal constraint in the state and give update rules based on the resets.

To eliminate $x - y \leq c$

States of $\mathcal{B}$: $(q, 0)$ and $(q, 1)$

For each transition of $\mathcal{A}$: $q \xrightarrow[R]{a,\ g} q'$

we have the following transitions in $\mathcal{B}$:

i) Suppose g does not contain $x - y \leq c$:

- Suppose neither x nor y is reset in R: $x \notin R$, $y \notin R$:

$$(q, b) \xrightarrow[R]{a,\ g} (q', b) \qquad b \in \{0, 1\}$$

- Suppose both x, y are reset: $x \in R$, $y \in R$.

$$(q, b) \xrightarrow[R]{a,\ g} (q', 1) \qquad \text{because } 0 \leq c$$

- Suppose y is reset in R, $x \notin R$:

$$(q, b) \xrightarrow[\{y\}]{a,\ g \wedge x \leq c} (q', 1)$$

$$(q, b) \xrightarrow[\{y\}]{a,\ g \wedge x > c} (q', 0)$$

Suppose $x \in R$, $y \notin R$:

$$(q, b) \xrightarrow[\{x\}]{a,\ g} (q', 1)$$

$$-y \leq c'$$

(ii)

Suppose g is of the form $g_1 \wedge \underline{x - y \leq c}$:

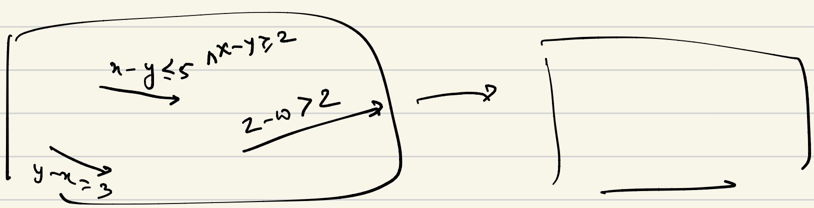
$$\cancel{(q, \underline{0}) \xrightarrow{a,\ g_1}}$$

Restrict to $(q, 1) \xrightarrow{a,\ g_1}$

Replace g with g_1 and do the same steps as before.

- We now have an automaton without $x - y \leq c$.

Question: What to do when there are multiple diagonal constraints?



~ You can eliminate one at a time.

In the final automaton without diagonal constraints:

$(q, \ 0 \ 1 \ 1 \ 0 \ 0)$

each bit is tracking one particular diagonal constraint

- This construction gives an exponential blow-up to the number of states.

- Is this blow-up unavoidable?

Diagonal constraints add exponential succinctness:

Construct a family of timed languages $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3 \dots$

$\{\mathcal{L}_n\}_{n \geq 1}$

s.t. For each $\mathcal{L}_n$ there is a d-timed automaton $\mathcal{A}_n$ with 2 states and n transitions.

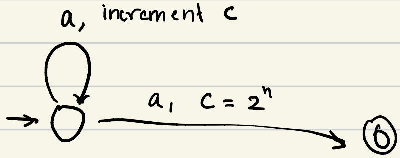
However, every diagonal-free automaton has at least 2^n states.

Language $\mathcal{L}_n$:

Simulate an 'n' bit counter:

0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
⋮

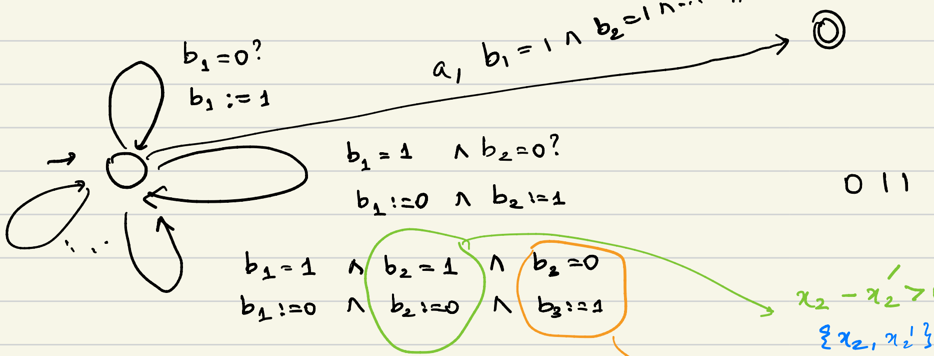
$$\{ (a^{2^n}, \tau) \mid \tau_1 < \tau_2 < \tau_3 \cdots < \tau_{2^n} \}$$



Add extra clock to ensure that between consecutive 'a's there is non-zero time elapse

c: a counter counting the no. of a's so far.

$c = b_{n-1}\, b_{n-2} \ldots\, b_2\, b_1$



→ only n transitions.

$x_3 - x_3' = 0$

$\{x_3'\}$

- How to do the above task using clocks?

- For each b_i: $\{x_i, x_i'\}$ $x_i - x_i' = 0$ denotes $b_i = 0$

  $x_i - x_i' > 0$ denotes $b_i = 1$

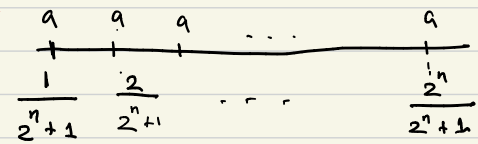
Final automaton: Automaton with counters as above,

with counters replaced with corresponding diagonal constraints and resets.

Every diagonal-free automaton for $\mathcal{L}_n$ requires at least 2^n states:

Suppose there exists a diagonal-free automaton $\mathcal{B}_n$ with $< 2^n$ states accepting $\mathcal{L}_n = \{ (a^{2^n}, \tau) \mid \tau_1 < \tau_2 \cdots < \tau_{2^n} \}$

w :



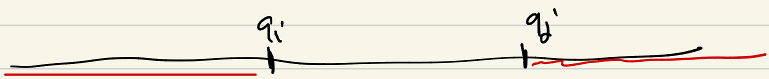
$\in \mathcal{L}_n$

$\mathcal{B}_n$ will have an accepting run on w.

$$(q_0, v_0) \xrightarrow{1/(2^n+1)} (q_1, v_1) \xrightarrow{1/(2^n+1)} (q_2, v_2) \longrightarrow \cdots \xrightarrow{1/(2^n+1)} (q_{2^n}, v_{2^n})$$

q_{2^n} is acc.

Since there are $< 2^n$ states, some $q_i = q_j$

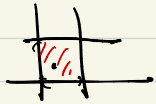


Claim: The run obtained by cutting the part between q_i and q_j is also a valid accepting run

$(q_0, v_0) \xrightarrow{1/2^n+1,\ g_0} (q_1, v_1) \xrightarrow{1/2^n+1,\ g_1} (q_2, v_2) \rightarrow \text{- - - -} \xrightarrow{g_i}$

v_0: 0 0 0 0 0 0 0

$\frac{1}{2^{n+1}}$ $\frac{1}{2^{n+1}}$ . . . $\frac{1}{2^{n+1}}$: g_0



Since g_0 is diagonal free it should allow

the open hypercube: $(0,1)^{|X|}$

- Similary g_2 should allow all valuation in $(0,1)^{|X|}$
- This property is true for every g_i in the run.

$\rightsquigarrow (q_i, v_i)$ $\quad (q_j, v_j) \xrightarrow{g_j} (q_{j+1}, v_{j+1})$

$\xrightarrow{1/2^{n+1},\ g_j}$

$(q_{j+1}, v'_{j+1}) \longrightarrow (q_{j+2}, v'_{j+2})$ - - - - (q_{2^n}, x'_{2^n})

A clock in v_i can have the foll. values: $\frac{i}{2^n+1}$, $\frac{i-1}{2^n+1}$ + ... , 0

- on time elapse of $\frac{1}{2^{n+1}}$: $\frac{i+1}{2^n+1}$, . . . $\frac{1}{2^{n+1}}$

Since g_j allows all valuations in $(0,1)^{|X|}$, it should allow $v_j + 1/2^{n+1}$.

- This way, we can extend the run till (q_{2^n}, v'_{2^n})

$\rightarrow$ A timed word with $< 2^n$ a's has an accepting run in $\mathcal{B}$.

$\rightarrow$ Contradiction!