TIMED AUTOMATA

LECTURE 17

GOALS OF TODAY'S LECTURE

- A partial algorithm for determinizing timed autl.
  - ↳ complete for several subclasses of T.A.

When are Timed Automata Determinizable?

Christel Baier[1], Nathalie Bertrand[2], Patricia Bouyer[3], and Thomas Brihaye[4]

[1] Technische Universität Dresden, Germany
[2] INRIA Rennes Bretagne Atlantique, France
[3] LSV, CNRS & ENS Cachan, France
[4] Université de Mons, Belgium

ICALP '09

Main idea: How to make a subset-construction work?

Recall the problem with subset construction:

$$q \xrightarrow[\{x\}]{a} q_1 \qquad q \xrightarrow{a} q_2 \qquad\qquad \{q\} \xrightarrow{a} \{q_1, q_2\}$$

How to track resets?

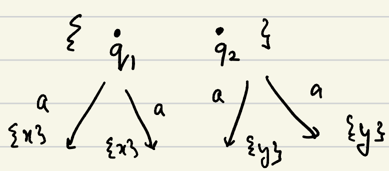
Suppose

$$q \xrightarrow[\{x\}]{a} q_1 \qquad q \xrightarrow[\{x\}]{a} q_2 \qquad\qquad \{q\} \xrightarrow[\{x\}]{a} \{q_1, q_2\}$$

No problem

Goal: 1. Convert the given NTA into a language equivalent B so that the same clock is reset on every transition with the same letter 'a' for every state 'q'.

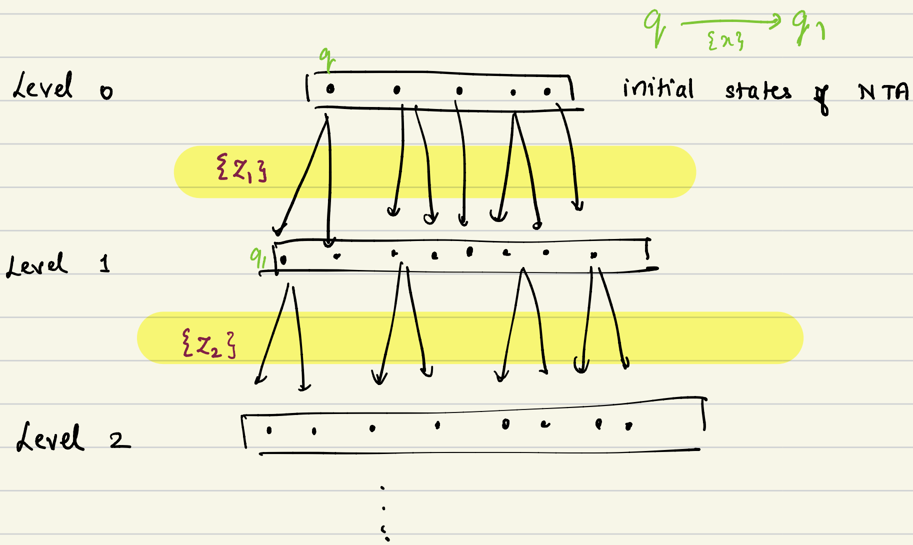
2. Suppose we try to do a subset construction



Even if same clock is reset out of q_1 and out of q_2, still there is a problem for the "subset".

To tackle these two problems, BBBB'09 gives a construction where

for each level → a new clock is reset



- This is the basic idea. There are two challenges
  - the constructed automaton should be language equivalent
  - it has to be finite.

Automaton $\mathcal{A}$: (Running example)

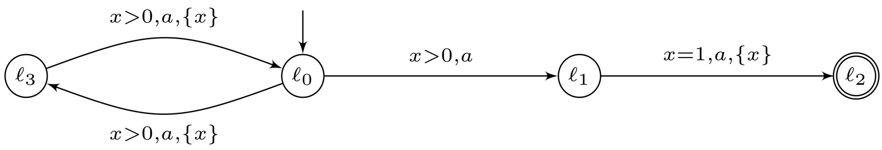
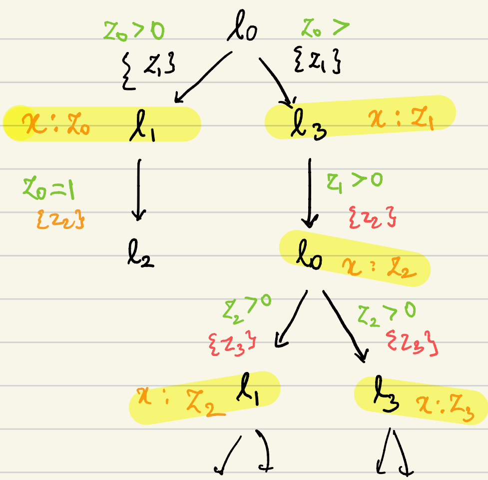


Fig. 1. A timed automaton $\mathcal{A}$

Step 1: $\mathcal{A}^\infty$: Infinite tree given by the unfolding of $\mathcal{A}$:

- a new clock is reset at each level.

$x: z_0$



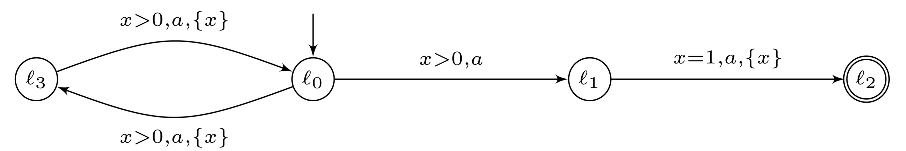
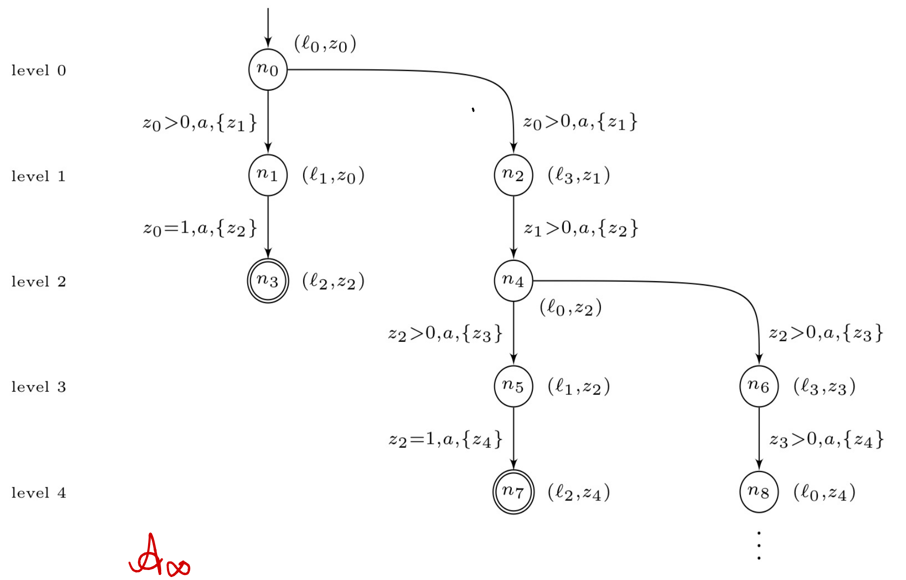


Fig. 1. A timed automaton $\mathcal{A}$



Advantages of $\mathcal{A}_\infty$: In a good form amenable to subset construction

Problems with $\mathcal{A}_\infty$: Infinitely many states.

Idea: If we know some $z_i > M$ (max constant), we can reuse it.

$\mathcal{A} \longrightarrow \mathcal{A}^{\infty}$:

Advantages: It is in a good form for subset construction

Problem: Infinite.

- We want to be able to "reuse" clocks

$\{z_i\}$

$\vdots$

$\{z_i\}$

Suppose we know that at a state, value of $z_i > M$

(M: maximal constant occurring in $\mathcal{A}$)

$x : z_i$ and $z_i > M$

$\downarrow$

$x : \perp$ and make z_i free.

To detect $z_i > M$, we will make use of a region-style construction.

Step 2: $R(\mathcal{A}_\infty)$: infinite timed automaton where guards are regions over relevant clocks.

Illustration:

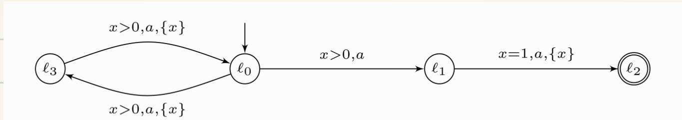


Fig. 1. A timed automaton $\mathcal{A}$

$(\ell_0, x : z_0), (z_0 = 0)$

$\cancel{z_0 = 0}$

Since $x > 0$ in $\ell_0 \to \ell_1$

$0 < z_0 < 1$, $\{z_1\}$ $\longrightarrow$ $(\ell_3, x : z_1)$, $0 < z_0 < 1$, $z_1 = 0$

$z_0 = 1$, $\{z_1\}$ $\longrightarrow$ $(\ell_3, x : z_1)$, $z_0 = 1$, $z_1 = 0$

$z_0 > 1$, $\{z_1\}$ $\longrightarrow$ $(\ell_3, x : z_1)$, $z_0 > 1$, $z_1 = 0$

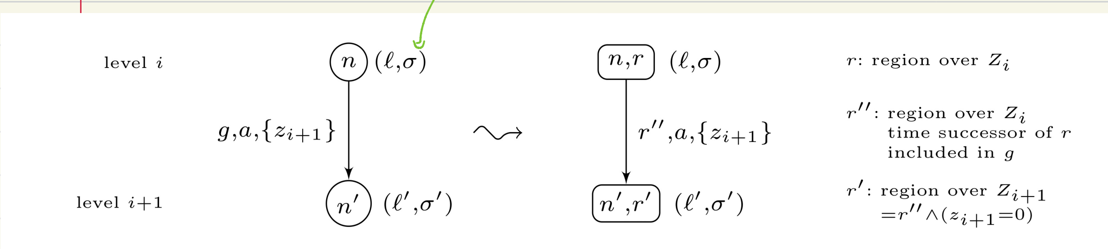
$0 < z_1 < z_0 < 1$, $\{z_2\}$

$0 < z_1 < z_0 = 1$, $\{z_2\}$

$0 < z_1 < 1 < z_0$, $\{z_2\}$

region over 3 clocks z_0, z_1, z_2

a map from $X \to \{z_0, \dots, z_i\}$



Currently we have an infinite tree whose nodes are:

Level i+1 $\quad \boxed{(\ell, \sigma), r}$

region over $\{z_0, \dots, z_i\}$

State of NTA

a map of clocks to $z_0, z_1, \dots, z_i$

$x_1 \mapsto z_{k_1}$

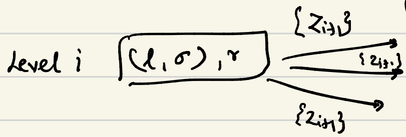
$x_2 \mapsto z_{k_2}$

$\vdots$

$x_n \mapsto z_{k_n}$

Advantages:

- 1. Amenable to subset construction since same resets on all outgoing transitions.



- 2. Node contains information about which clocks are above M.

Problems: - Still infinite

To make it finite we need a facility to reuse clocks.
As a first step, we restrict regions to "active clocks".

Active clocks:

A clock z_i is active at node $(\ell, \sigma), r$ if

- 1. Some clock 'x' is mapped to z_i AND
- 2. $z_i \leq M$ in the region 'r'.

- First of all, note that if some clock z_j is not mapped to any clock in σ, then its value is useless for the future. So it can be removed.
- Secondly, if $\sigma(x) = z_i$ and $z_i > M$ in r, then we can maintain this information in σ itself.

$$\sigma(x) = \perp \quad (\text{to denote that its value} > M)$$

Once again, such a z_i can be removed.

We can therefore assume that the regions in $R(A^\infty)$ are only over active clocks!

We now perform a subset construction on this region tree.

Subset construction:

Node:

$$\boxed{\{(\ell_1, \sigma_1), (\ell_2, \sigma_2) \dots (\ell_k, \sigma_k)\}, \gamma}$$

a subset of state-map pairs

a region. over active clocks.

Transitions: Look at all a-transitions from:

$$(\ell_1, \sigma_1), \gamma \qquad (\ell_2, \sigma_2), \gamma \quad \dots \quad (\ell_k, \sigma_k), \gamma$$

$$\downarrow \gamma' \qquad\qquad \downarrow \gamma' \qquad\qquad \downarrow \gamma'$$

- from the same time successor γ'.
- Pick a fresh clock z which is not present in γ

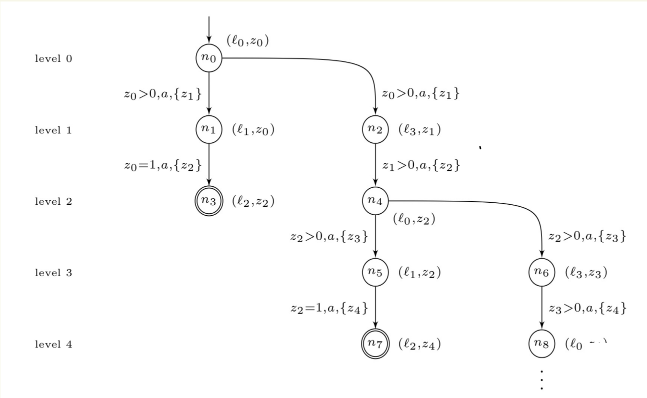
$$\{(\ell_1, \sigma_1), (\ell_2, \sigma_2) \dots (\ell_k, \sigma_k)\}, \gamma$$

$$\Big\downarrow \gamma'$$

$$\{(\ell_1', \sigma_1')(\ell_1'', \sigma_1'') \dots (\ell_k', \sigma_k')\}, \gamma''$$

over active clocks.

Illustration of Step 3:

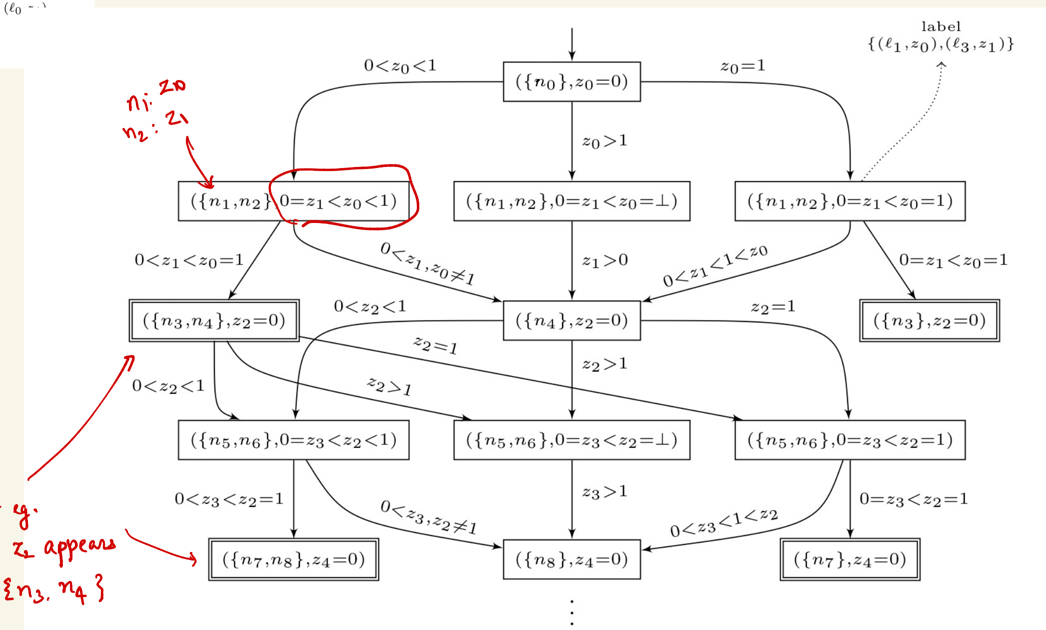


Symbolic determinization of $R(A^\infty)$.

$$A \xrightarrow{\text{new reset}} A^\infty \xrightarrow[\text{regions}]{\text{added}} R(A^\infty) \xrightarrow[\text{restrict to clocks that are present in the map}]{\text{subset}} \mathrm{SymbDet}(R(A^\infty))$$

$$R(A^\infty) \longrightarrow \mathrm{SymbDet}(R(A^\infty))$$

$n_1 : z_0$
$n_2 : z_1$



restricted to clocks which appear in a map of the subset

for eg. only z_2 appears in $\{n_3, n_4\}$

Final observation: If the number of active clocks is bounded by γ, we can reuse clocks and get a finite timed automaton.

$X = \{x_1, x_2, \dots x_\gamma\}$

based on some deterministic policy.

One more optimization:

- If the successor is due to a "zero-time" successor of the region, then use the previously reset clock (in the transition that lead to this region).

$\{z\}$

$\{(\ell_1, \sigma_1) \dots (\ell_k, \sigma_k)\}, \gamma$ $\qquad \gamma \models z = 0$

r' [0-time successor, that is: $r' = r$]

$\{z\}$

$\mathcal{A}$ $\{y_1, y_2 \cdots y_n\}$

$\downarrow$

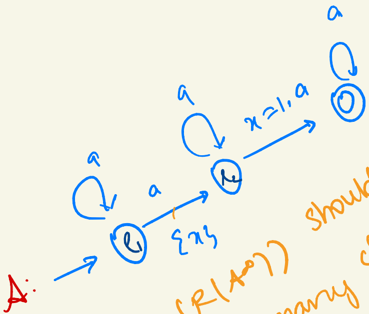
$\mathcal{A}^\infty$ | $(\ell, y_1 : z_i, y_2, z_{i_2} \cdots)$

$\downarrow$

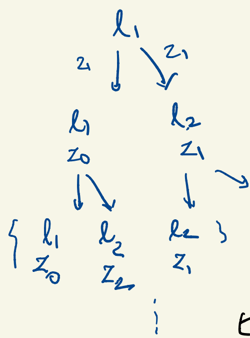
$R(\mathcal{A}^\infty)$ | Every node is augmented with a region involving all clocks above its level.

$\downarrow$

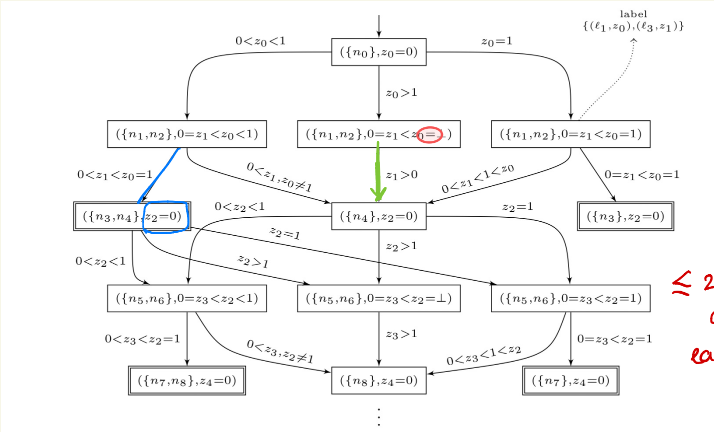
$\boxed{\mathrm{SymbDet}(R(\mathcal{A}^\infty)) :}$ subset construction restricted to clocks that appear in the maps.



SymbDet $(R(\mathcal{A}^\infty))$ should not have finitely many clocks.



If the number of clocks present in each region is finite then there is a way in which we can extract an equivalent D.T.A. using a fresh set of clocks $\{x_1, \ldots x_\gamma\}$



≤ 2 active clocks in each node.

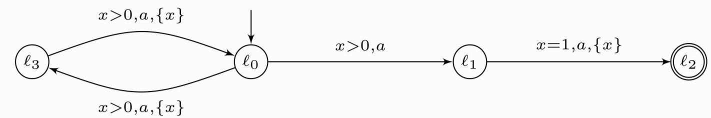
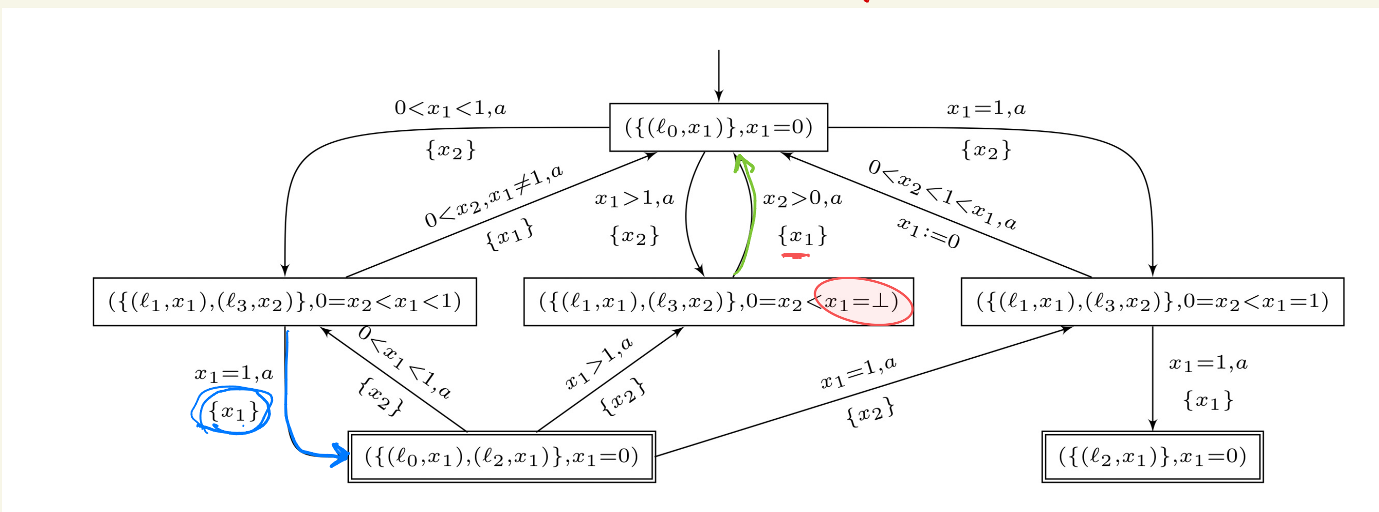


Fig. 1. A timed automaton $\mathcal{A}$

$\{x_1, x_2\}$

with active clock reduction + reuse of clocks.



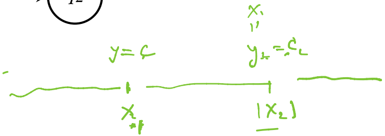
Question:

- What are some sufficient conditions for an automaton to have only finitely many active clocks? That is, finite γ.

Coming next: Two subclasses of NTA for which this construction works.

Integer reset timed automata

$$q_1 \xrightarrow[R]{a,\ g} q_2$$



Conditions:

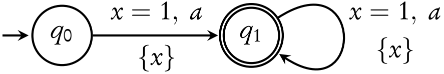
- g has **integer** constants
- R is **non-empty** $\Rightarrow$ g has some constraint $x = c$

Implication:

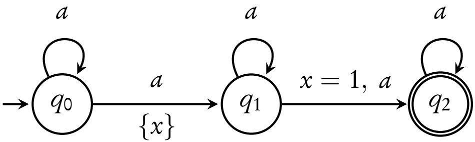
- Along a timed word, a **reset** of an IRTA happens only at **integer timestamps**

Timed automata with integer resets: Language inclusion and expressiveness

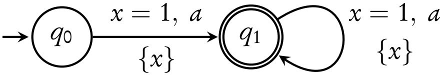
Suman, Pandya, Krishna, Manasa. *FORMATS'08*



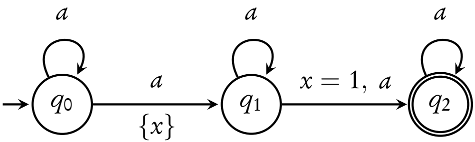
an IRTA



not an IRTA



an IRTA



not an IRTA

Next: **determinizing** IRTA using the **subset construction**

$$\{(\ ), (\ ), (\ ), \ldots (\ )\}, r \xrightarrow{\{z\}} \{(\ )(\ )(\ )\ldots(\ )\}, r'$$

- Suppose a new clock z becomes active in the successor
- This means some clock was reset in the previous transition
- Moreover, due to the "zero-time" optimization, the new clock is added only for a non-zero delay.
- Therefore > 0 time should have elapsed from 'r'.
- Due to property of IRTA, this non-zero delay should be an integer ≥ 1.
- If there are M+1 active clocks in a node, then starting from the oldest active clock 'y' in this node, at least M time has elapsed.

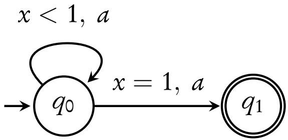
  Therefore 'y' will become inactive (> M) and can be reused.

M+1 active clocks are sufficient for IRTA.

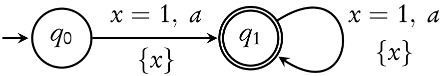
Strongly non-Zeno automata

A TA is strongly non-Zeno if there is $K \in \mathbb{N}$:

every sequence of greater than K transitions **elapses** at least **1 time unit**



not SNZ



SNZ

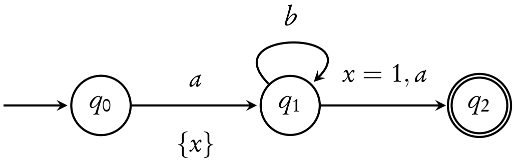
$K=2$

a a a . . .

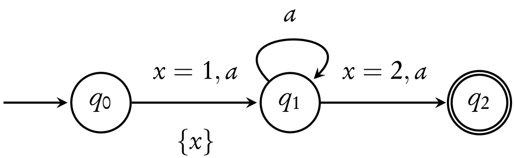
1 2 3

- In a SNZ automaton, every sequence of $K(M+1)$ transitions elapses $> M$ time units.

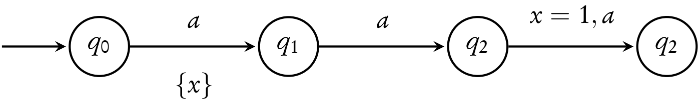
- Therefore when there $K(M+1)$ active clocks in a node, the oldest entrant will become inactive and can be reused again.



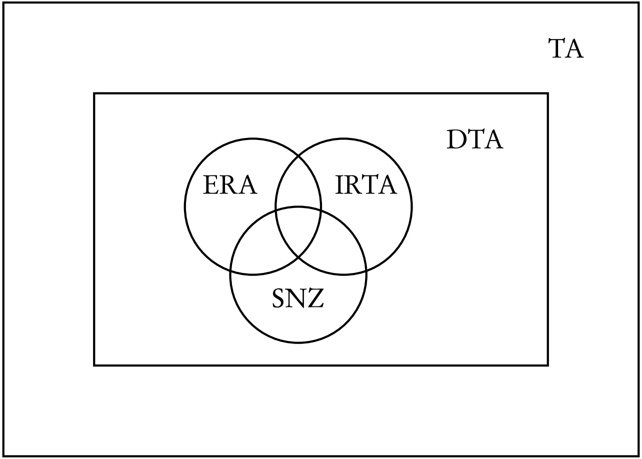
ERA ~~IRTA~~ ~~SNZ~~



~~ERA~~ IRTA ~~SNZ~~



~~ERA~~ ~~IRTA~~ SNZ


TA
DTA
ERA
IRTA
SNZ

Summary:

- A method to determinize NTA, that works for certain classes.

- New subclasses seen: IRTA
  SNZ.

- For a generic $\mathcal{A}$, this algorithm may not work.

Can we decide, given $\mathcal{A}$, whether the no. of active clocks in SymbDet $(R(\mathcal{A}^*))$ is finite?

→ Given $\mathcal{A}$, can we decide whether this algorithm will work for this automaton or not?

Given NTA $\mathcal{A}$. does there exist a language equiv. DTA.

↳ Undecidable.