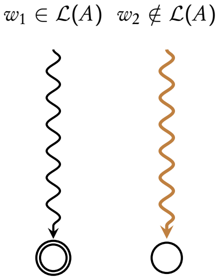
Theorem

Deterministic timed automata are **closed under complement**

Theorem

Deterministic timed automata are **closed under complement**

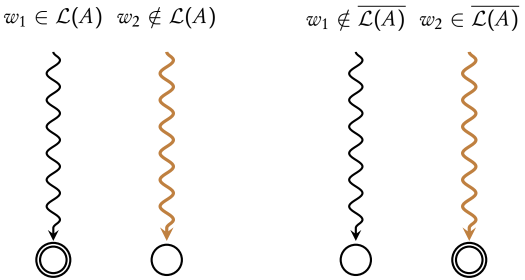
1. **Unique** run for every timed word



Theorem

Deterministic timed automata are **closed under complement**

1. **Unique** run for every timed word
2. **Complementation: Interchange** acc. and non-acc. states

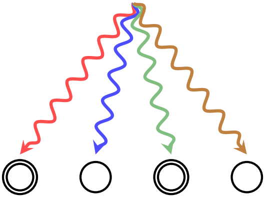


Theorem (Lecture 1)

Non-deterministic timed automata are **not closed under complement**

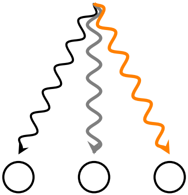
Many runs for a timed word

$w_1 \in \mathcal{L}(A)$



Exists an acc. run

$w_2 \notin \mathcal{L}(A)$



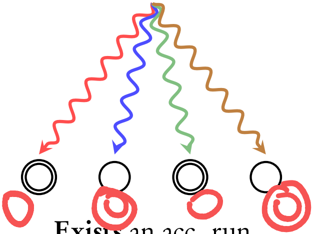
All runs non-acc.

Theorem (Lecture 1)

Non-deterministic timed automata are **not closed under complement**

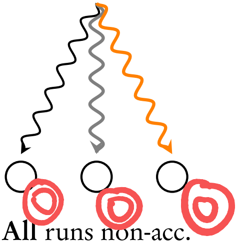
Many runs for a timed word

$w_1 \in \mathcal{L}(A)$



Exists an acc. run

$w_2 \notin \mathcal{L}(A)$



All runs non-acc.

Complementation: **interchange** acc/non-acc + ask are **all runs acc.** ?

A timed automaton model with **existential** and **universal** semantics for acceptance

Alternating timed automata

Lasota and Walukiewicz. *FoSSaCS'05*, *ACM TOCL'2008*

Section 1:

Introduction to ATA

- X : set of **clocks**
- $\Phi(X)$: set of clock constraints σ (**guards**)

$$\sigma : \; x < c \mid x \leq c \mid \sigma_1 \wedge \sigma_2 \mid \neg\sigma$$

  c is a non-negative **integer**

- Timed automaton A: $(Q, Q_0, \Sigma, X, T, F)$

$$T \;\subseteq\; Q \times \Sigma \times \Phi(X) \times Q \times \mathcal{P}(X)$$

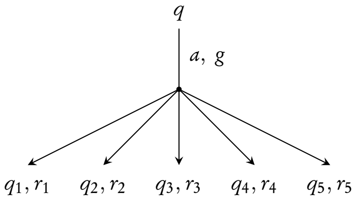
$$q \xrightarrow{a,\ g} q'$$

R

$$T \subseteq Q \times \Sigma \times \Phi(X) \times Q \times \mathcal{P}(X)$$

$$\downarrow$$

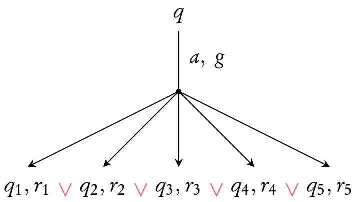
$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X))$$



$$T \subseteq Q \times \Sigma \times \Phi(X) \times Q \times \mathcal{P}(X)$$

$$\Big\downarrow$$

$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X))$$



$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X))$$

$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X))$$

$$\Big\downarrow \; \mathcal{B}^+(S) \text{ is all } \phi ::= S \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2$$

$$T : Q \times \Sigma \times \Phi(X) \;\mapsto\; \mathcal{B}^+(Q \times \mathcal{P}(X))$$

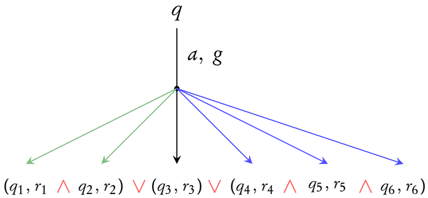
$$Q \times P(X) = \{ (q_1, r_1), (q_2, r_2), \vdots, q_n, r_n) \}$$

$$\left[(q_1, r_1) \vee (q_2, r_2) \right] \wedge (q_3, r_3)$$

$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{P}(Q \times \mathcal{P}(X))$$

$$\Big\downarrow \quad \mathcal{B}^+(S) \text{ is all } \phi ::= S \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2$$

$$\boxed{T : Q \times \Sigma \times \Phi(X) \;\mapsto\; \mathcal{B}^+(Q \times \mathcal{P}(X))}$$



Alternating Timed Automata

An **ATA** is a tuple $A = (Q, q_0, \Sigma, X, T, F)$ where:

$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{B}^+(Q \times \mathcal{P}(X))$$

is a **finite partial function.**

Alternating Timed Automata

An **ATA** is a tuple $A = (Q, q_0, \Sigma, X, T, F)$ where:

$$T : Q \times \Sigma \times \Phi(X) \mapsto \mathcal{B}^+(Q \times \mathcal{P}(X))$$

is a **finite partial function.**

$\Phi(x) = \{ g_1, g_2, \ldots \}$

Partition: For every q, a the set

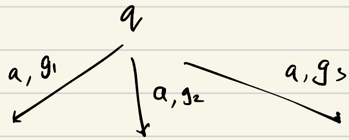
(q, a, g_1)

(q, a, g_2)

$$\{ [\sigma] \mid T(q, a, \sigma) \text{ is defined} \}$$

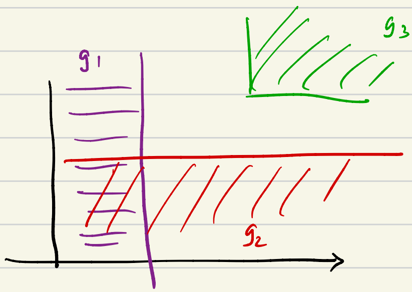
~~(q, a, g_3)~~

gives a finite partition of $\mathbb{R}_{\geq 0}^X$

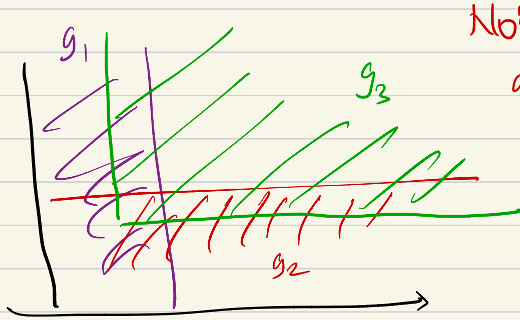


$g_1 \cup g_2 \cup g_3$ gives all valuations

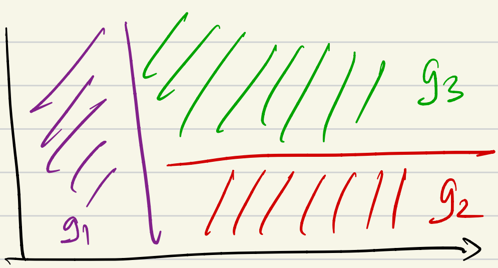
$X = \{x, y\}$



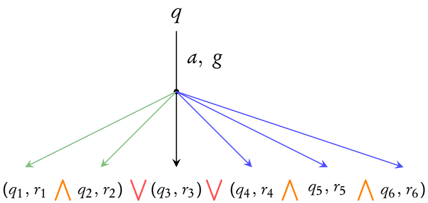
Not allowed.



Not allowed as not a partition

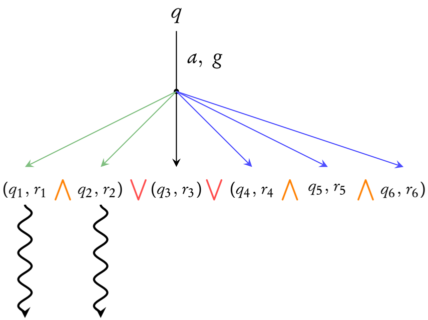


Acceptance



Accepting run from q iff:

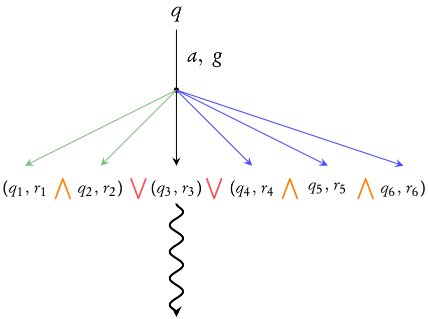
Acceptance



Accepting run from q iff:

- accepting run from q_1 **and** q_2,

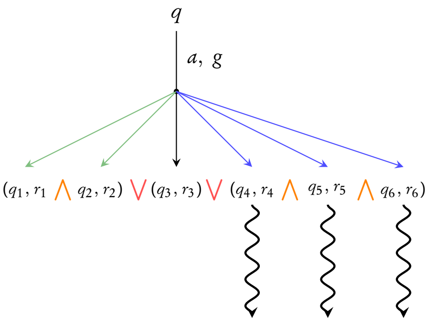
Acceptance



Accepting run from q iff:

- accepting run from q_1 **and** q_2,
- **or** accepting run from q_3,

Acceptance

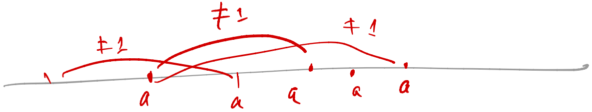


Accepting run from q iff:

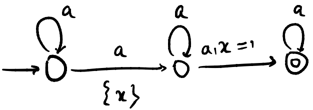
- accepting run from q_1 **and** q_2,
- **or** accepting run from q_3,
- **or** accepting run from q_4 **and** q_5 **and** q_6

L : timed words over $\{a\}$ containing **no two** $a's$ at distance 1

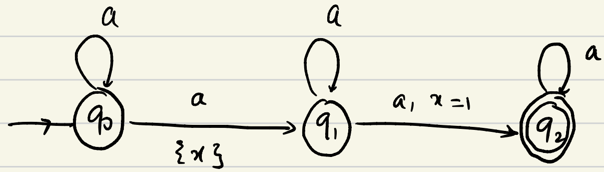
(Not expressible by non-deterministic TA)



Complement of L: ∃ 2 a's at distance 1 apart.



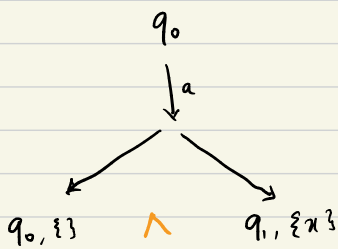
I



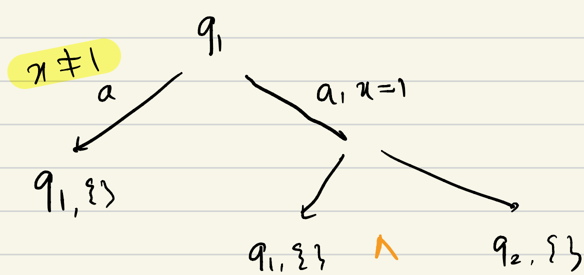
Non-deterministic TA.

ATA for C

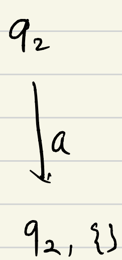
T1.



T2.

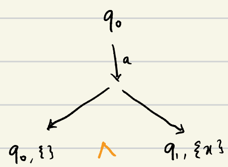


T3.

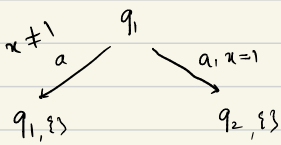


Acc. state $\{q_0, q_1\}$

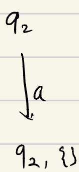
T1.



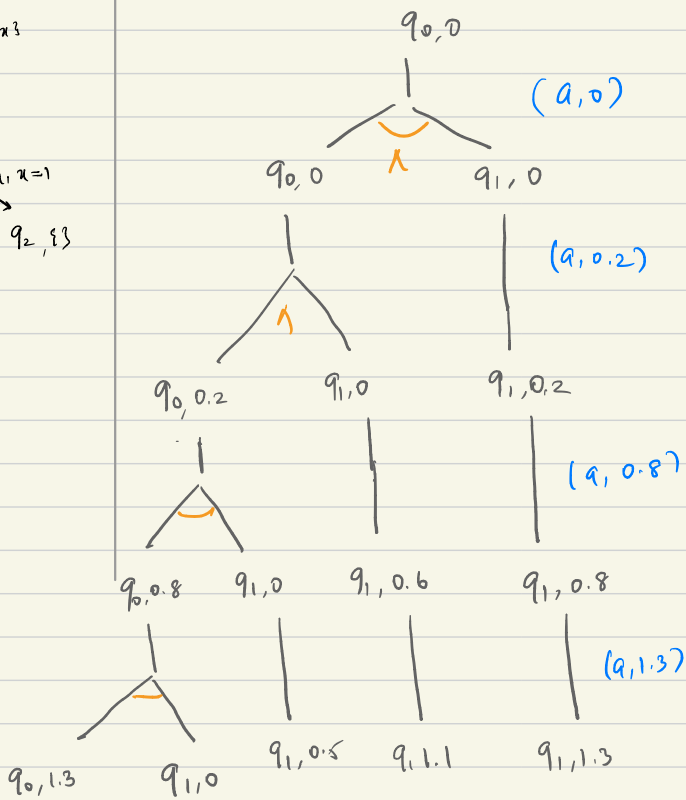
T2.



T3.

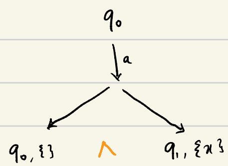


(a, 0) (a, 0.2) (a, 0.8) (a, 1.3)

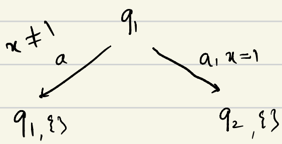


Accepting

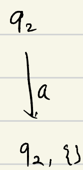
T1.



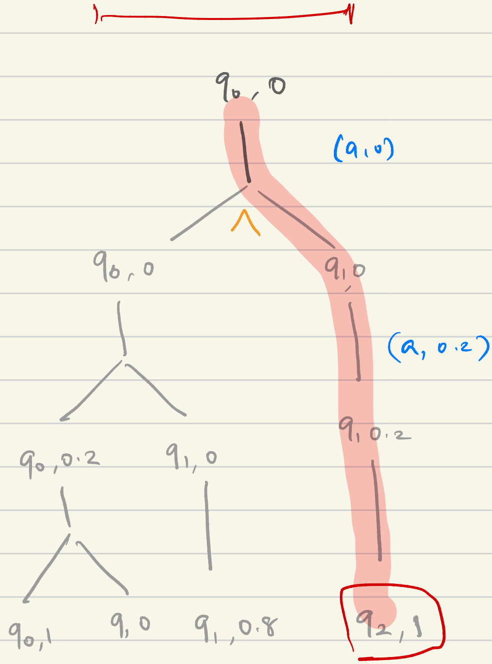
T2.



T3.



(a, 0) (a, 0.2) (a, 1)



Witness for non-acceptance

L : timed words over $\{a\}$ containing **no two** $a's$ at distance 1

(Not expressible by non-deterministic TA)

ATA:

$$q_0, a, tt \mapsto (q_0, \emptyset) \wedge (q_1, \{x\})$$

$$q_1, a, x = 1 \mapsto (q_2, \emptyset)$$

$$q_1, a, x \neq 1 \mapsto (q_1, \emptyset)$$

$$q_2, a, tt \mapsto (q_2, \emptyset)$$

q_0, q_1 are acc., q_2 is non-acc.

Acceptance Game: $\mathcal{G}_{A,w}$

$$w := (a_0, t_0)\,(a_1, t_1)\,(a_2, t_2)\,\dots\,(a_{k+1}, t_{k+1})\,\dots\,(a_n, t_n)$$

Phase 0 Phase 1 Phase k+1

(q_k, v_k)

$$\bar{v} = v_k + t_{k+1} - t_k$$

$$q_k \xrightarrow{a_{k+1},\, g_1} \qquad q_k \xrightarrow{a_{k+1},\, g_2} \qquad \vdots \qquad q_k \xrightarrow{a_{k+1},\, g_m}$$

- Let σ be unique constraint s.t. $\bar{v}$ satisfies σ

$$b = \delta(q_{k+1}, a_{k+1}, \sigma)$$

- $b = b_1 \wedge b_2$: Adam chooses a subformula and game continues with the subformula.

- $b = b_1 \vee b_2$: Eve

- $b = (q, r) \in Q \times \mathcal{P}(C)$

$((q_1, r_1) \vee (q_2, r_2)) \wedge (q_3, r_3)$

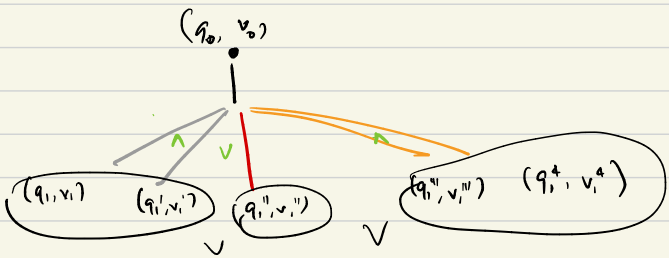
- Phase ends with

$$(q_{k+1}, v_{k+1}) := (q, \bar{v}[r := 0])$$

$\longrightarrow$ Play ends with (q_{n+1}, v_{n+1})

- Eve wins the play if q_{n+1} is accepting; otherwise Adam wins.

- $w \in \mathcal{L}(\mathcal{A})$ if Eve has a strategy to win $\mathcal{G}_{\mathcal{A}, w}$. Else $w \notin \mathcal{L}(\mathcal{A})$.



Acceptance game $\mathcal{G}_{A,w}$:

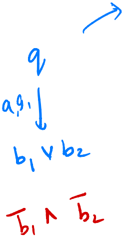
$$\mathcal{L}(A) = \{ w \mid \text{Eve wins } \mathcal{G}_{A,w} \}$$

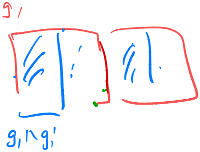
Summary:

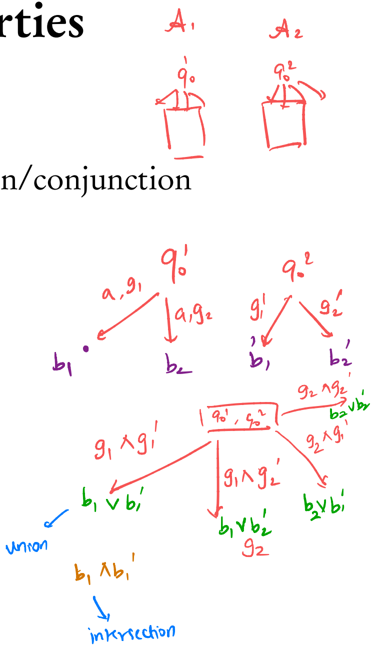
- A model involving existential and universal transitions.
  ↳ Alternating T.A.
- 1 Example
- Acceptance game $\mathcal{G}_{A,w}$.

Closure properties

- Union, intersection: use disjunction/conjunction
- Complementation: **interchange**
  1. acc./non-acc.
  2. conjunction/disjunction







Complementation:

$$q \xrightarrow{a,\, g_1}$$

$\vee$

$$[(q_1, r_1) \wedge (q_2, r_2)] \vee (q_3, r_3) \vee [(q_4, r_4) \wedge (q_5, r_5)]$$

$\downarrow$ complementation:

$$q \xrightarrow{a,\, g_1}$$

$$[(q_1, r_1) \vee (q_2, r_2)] \wedge (q_3, r_3) \wedge [(q_4, r_4) \vee (q_5, r_5)]$$

Closure properties

- Union, intersection: use disjunction/conjunction
- Complementation: **interchange**
  1. acc./non-acc.
  2. conjunction/disjunction

No change in the number of clocks!

Section 2:

The 1-clock restriction

- Emptiness: given A, is $\mathcal{L}(A)$ empty
- Universality: given A, does $\mathcal{L}(A)$ contain all timed words
- Inclusion: given A, B, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$

- Emptiness: given A, is $\mathcal{L}(A)$ empty
- Universality: given A, does $\mathcal{L}(A)$ contain all timed words
- Inclusion: given A, B, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$

Undecidable for **two clocks or more** (~~via Lecture 3~~)

Universality problem for NTA $\longrightarrow$ Emptiness problem for ATA

$\mathcal{B}$ $\longrightarrow$ Look at $\mathcal{B}$ as an ATA.

Complement it and check for emptiness

$\mathcal{L}(\mathcal{B})$ is universal iff

$\overline{\mathcal{L}(\mathcal{B})}$ is empty

- Emptiness: given A, is $\mathcal{L}(A)$ empty
- Universality: given A, does $\mathcal{L}(A)$ contain all timed words
- Inclusion: given A, B, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$

Undecidable for **two clocks or more** (~~via Lecture 3~~)

Decidable for **one clock** (~~via Lecture 4~~)

- Emptiness: given A, is $\mathcal{L}(A)$ empty
- Universality: given A, does $\mathcal{L}(A)$ contain all timed words
- Inclusion: given A, B, is $\mathcal{L}(A) \subseteq \mathcal{L}(B)$

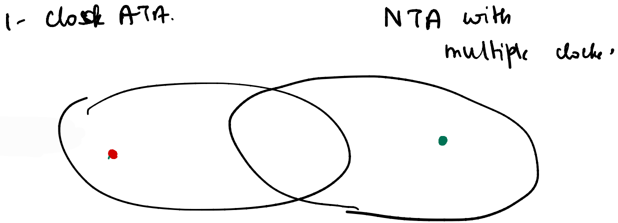
Undecidable for **two clocks or more** (~~via Lecture 3~~)

Decidable for **one clock** (~~via Lecture 4~~)

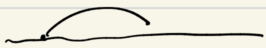
Restrict to one-clock ATA

Theorem

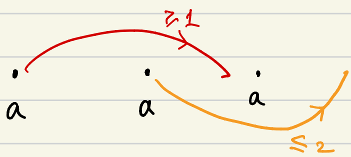
Languages recognizable by 1-clock ATA and (many clock) TA are **incomparable**



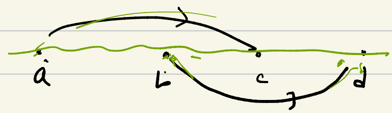
Alternation vs. Multiple clocks:

Alternation:  For every point, ∃ another at distance c.

Multiple clocks:



Interleaving.



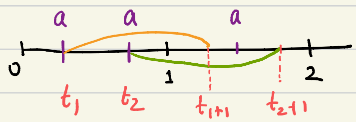
Need multiple clocks

Example L: no a's at distance 1. → 1-clock ATA.
but no NTA.

Example of a language accepted using multiple clock T.A.,
but not 1-ATA?

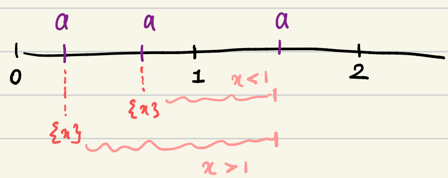
Clarification about the expressive power of 1-ATA:

Question: Let $L_1 = \{ (aaa,\ t_1\, t_2\, t_3) \mid 0 < t_1 < t_2 < 1,\ t_1 + 1 < t_3 < t_2 + 1 \}$



Can you construct a 1-ATA for L_1?

Idea:



1-ATA:

$(q_0, a, 0 < x < 1) \longmapsto (p_1, \{x\}) \wedge (s_1, \emptyset)$

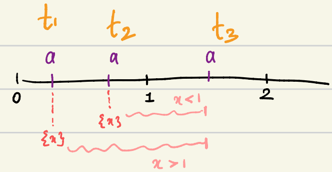
$x>0$ $(p_1, a, \cancel{true}) \longmapsto (p_2, \emptyset)$

$(s_1, a, x < 1) \longmapsto (s_2, \{x\})$

$(p_2, a, x > 1) \longmapsto (f, \emptyset)$

$(s_2, a, x < 1) \longmapsto (f, \emptyset)$

$(f, a, true) \longmapsto (reject, \emptyset)$, $(reject, a, true) \longmapsto (reject, \emptyset)$



1-ATA: $(q_0, a, 0<x<1) \longmapsto (p_1, \{x\}) \wedge (s_1, \phi)$

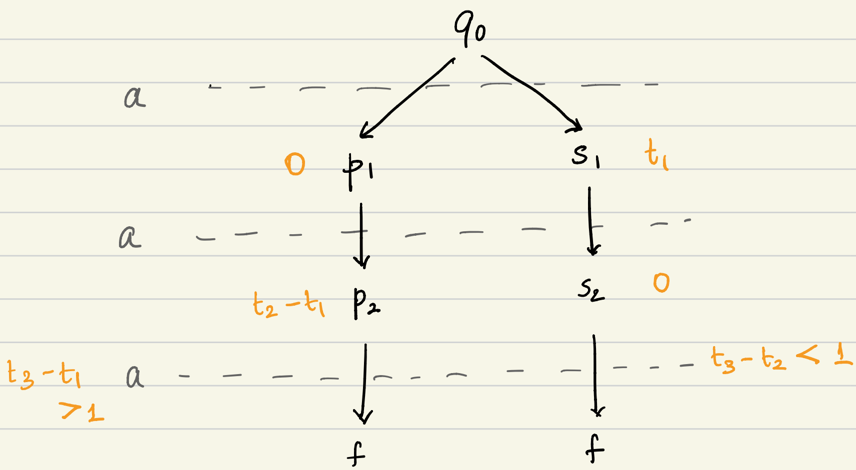
$(p_1, a, \cancel{true}\ x>0) \longmapsto (p_2, \phi)$

$(s_1, a, x<1) \longmapsto (s_2, \{x\})$

$(p_2, a, x>1) \longmapsto (f, \phi)$

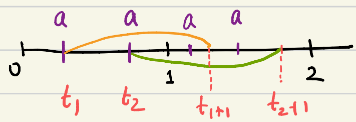
$(s_2, a, x<1) \longmapsto (f, \phi)$

$(f, a, true) \longmapsto (reject, \phi)$, $(reject, a, true) \longmapsto (reject, \phi)$

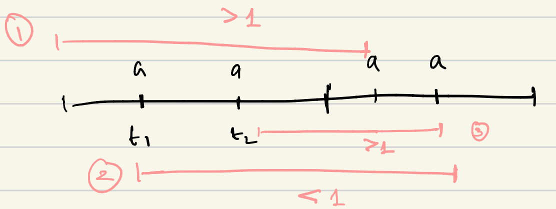


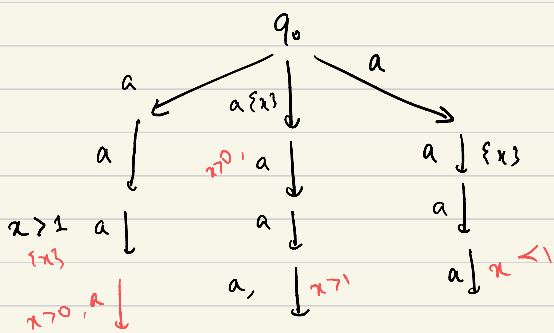
A small modification of the previous example:

Question: Let $L_2 = \{ (aaaa, t_1 t_2 t_3 t_4) \mid 0 < t_1 < t_2 < 1, \; 1 < t_3 < t_4, \; t_1 + 1 < t_4 < t_2 + 1 \}$



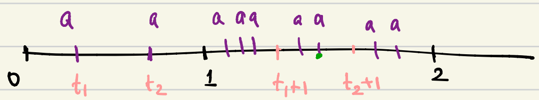
Can you construct a 1- ATA for L_2?



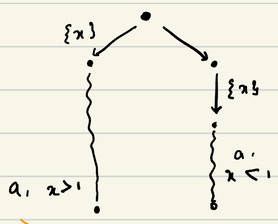


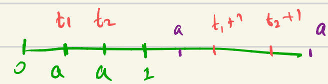
Question:

$$\mathcal{L}_3 = \{ (a^k, t_1 t_2 \dots t_k) \mid k \geq 3 :$$
$$0 < t_1 < t_2 < 1$$
$$\exists j \geq 3 \text{ s.t. } t_1 + 1 < t_j < t_2 + 1 \quad \}$$
$$t_3 > 1$$

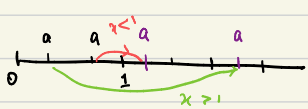


Problem:





The two a's could be different

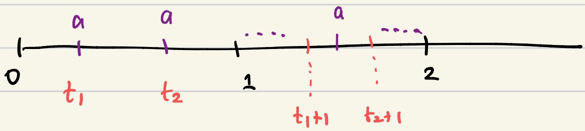


- This is an intuition that $\mathcal{L}_3$ cannot be accepted by a 1-ATA.
- However, proving that a language cannot be accepted by a 1-ATA is difficult.
- We will see another example given in the paper, for which there is a proof that it cannot be accepted by a 1-ATA.

$L = \{ (a^k, t_1 t_2 \dots t_k) \mid 0 < t_1 < t_2 < 1$

$1 < t_3, \dots, t_k < 2$

there is exactly one a between $t_1 + 1$ and $t_2 + 1$ $\}$



→ L can be accepted by a deterministic T.A with 2 clocks.

Goal: To prove that L cannot be accepted by a 1-ATA.

Step 1: Understand some property of DFAs

Step 2: How Step 1 translates to untimed alternating finite automata

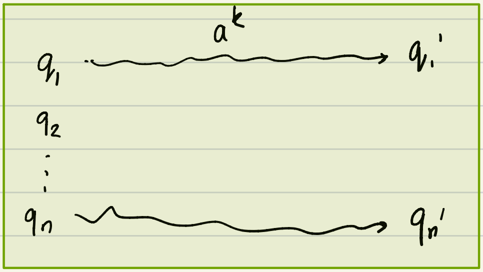
Step 3: Any 1-ATA accepting L behaves like an untimed AFA in the interval $(1, 2)$, where clocks are useless.

Step 4: Use Step 1 and 2 in 3 to get a contradiction.

Step 1: Understanding a property of DFAs.

- Consider a unary alphabet $\{a\}$, and DFA $\mathcal{B} = (Q, q_0, \delta, F)$
- For each a^k, the DFA gives rise to a function

$$f_k^{\mathcal{B}} : Q \longmapsto Q$$



- The number of functions from $Q \longmapsto Q$ is finite.
- Therefore, if we look at the sequence:

$$f_1^{\mathcal{B}}, \quad f_2^{\mathcal{B}}, \quad f_3^{\mathcal{B}}, \ldots\ldots .$$

there exist m, ℓ, s.t.

$$\boxed{f_m^{\mathcal{B}} = f_{m+\ell}^{\mathcal{B}}}$$

f_m | $f_{m+\ell}$

$q_1 — q_1'$ | $q_1 — q_1'$

$q_2 — q_2'$ | $q_2 — q_2'$

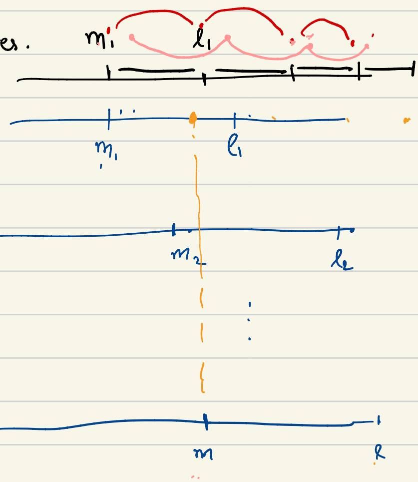
f_{m+1} = $f_{m+\ell+1}$

$q_1 \longrightarrow \delta(q_1', a)$

$q_2 — \delta(q_2', a)$

- Moreover: $\quad f_{m+i}^{\mathcal{B}} = f_{m+\ell+i}^{\mathcal{B}} \quad \forall i \geq 0$

Consider all DFA with atmost n states.



finitely many

$$\left\{\begin{array}{lll} B_1 & - & m_1, l_1 \\ B_2 & - & m_2, l_2 \\ \vdots & \vdots & \\ B_j & - & m_j, l_j \end{array}\right.$$

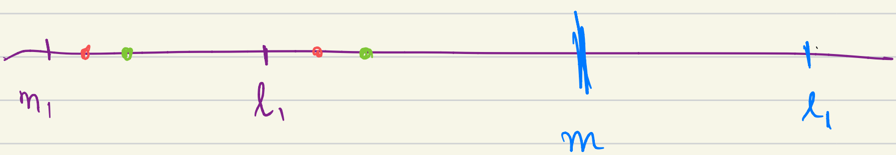
- Let $m = \max(m_1, \dots m_j)$

$l = l_1 \cdot l_2 \cdot l_3 \cdots l_j$

Then for every DFA B with $\leq n$ states, we have:

$$f^{B}_{m+i} = f^{B}_{m+l+i} \qquad \forall i \geq 0$$

$f_{m+i} = f_{m+kl_1+i}$

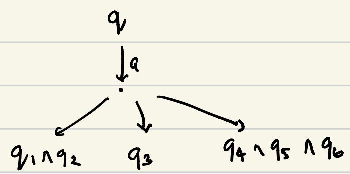


$$f_{m_1+i} = f_{m_1+l_1+i} = f_{m_1+2l_1+i} = f_{m_1+3l_1+i}$$

Step 2: Translating Step 1 to alternating finite automata.

AFA: (Q, q_0, δ, F)

$$\delta : Q \times \Sigma \longmapsto \mathcal{B}^+(Q)$$



Syntax and semantics similar to ATA: with no guards, no resets

Claim: Every AFA can be converted into an equivalent DFA.

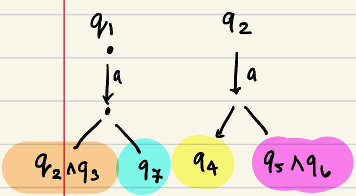
Modified subset construction:

Each node: a set of subsets of Q

$$\{\{q_1, q_2\}, \{q_1, q_3, q_4\}, \{q_2, q_5\}, \{q_3\}\}$$

$$\downarrow a$$

?



$$\{q_1, q_2\}$$

$$\downarrow a$$

$$\{\{q_2, q_3, q_4\}, \{q_2, q_3, q_5, q_6\}, \{q_7, q_4\}, \{q_7, q_5, q_6\}\}$$

- Perform the above operation on each set from the set of subsets.

- Node is accepting if there is a subset containing only accepting states.

Theorem: Every AFA with 'n' states can be converted into a DFA with $\leq 2^{2^n}$ states.

Consider unary alphabet $\{a\}$.

An AFA $\mathcal{A}$ with state set Q gives a function :

$$\boxed{f_k^{\mathcal{A}} : 2^{2^Q} \longmapsto 2^{2^Q}}$$

$$\{\{\ \}, \{\ \}, \{\ \} \dots \{\ \}\} \xrightarrow{a^k} \{\{\ \}, \{\ \}, \{\ \} \dots \{\ \}\}.$$

$$\vdots$$

$$\{\{\ \}, \{\ \}\} \xrightarrow{a^k} \{\{\ \}, \{\ \}, \{\ \}\}.$$

- Similar to the DFA case, let 'm', 'ℓ' be numbers s.t.

$$f_{m+i}^{\mathcal{A}} = f_{m+\ell+i}^{\mathcal{A}} \quad \forall i \geq 0$$

for all AFA $\mathcal{A}$ with atmost $2n$ states

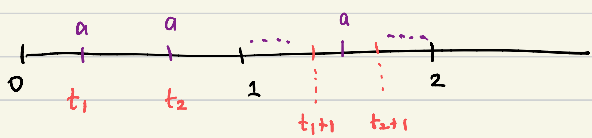
- Starting from some $\{q\}$, a^{m+i} goes to an accepting node iff $a^{m+\ell+i}$ goes to an accepting node.

Recall:

$$L = \{ (a^k, t_1 t_2 \dots t_k) \mid 0 < t_1 < t_2 < 1$$

$$1 < t_3, \dots, t_k < 2$$

there is exactly one a between $t_1 + 1$ and $t_2 + 1$ $\}$

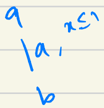


Step 1: Understand some property of DFAs

Step 2: How Step 1 translates to untimed alternating finite automata

Step 3: Any 1-ATA accepting L behaves like an untimed AFA in the interval $(1, 2)$, where clocks are useless.

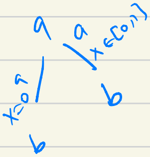
Step 4: Use Step 1 and 2 in 3 to get a contradiction.



Suppose $\mathcal{A}$ is a 1-ATA with 'n' states accepting 'L'.

- We can assume that every transition is partitioned as:

$$x = 0 \quad \Big| \quad 0 < x < 1 \quad \Big| \quad x = 1 \quad \Big| \quad 1 < x < 2 \quad \Big|$$ beyond 2 original guard.

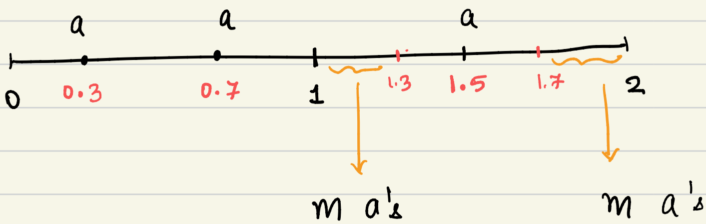


- For the moment, let us ignore all transitions with $x = 0$. We will see later why we can do this.

Construct two timed words w_1 and w_2 as follows:

No two 'a's come at the same time-stamp

w_1 :



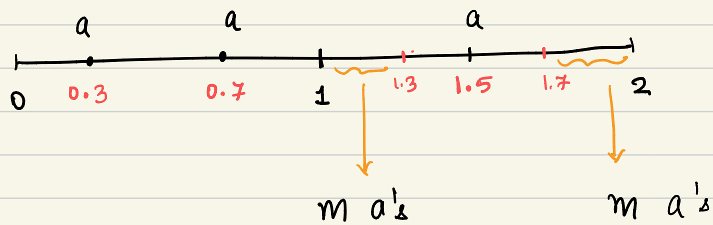
(m, ℓ are as chosen before)

w_2 : On top of w_1, add 'ℓ' a's in the interval $(1.3, 1.7)$, but not at 1.5

$w_1 \in L$, $\qquad w_2 \notin L$.

We will show that if $\mathcal{A}$ accepts w_1, it also accepts w_2 – a contradiction.

w_1 :



(m, l are as chosen before)

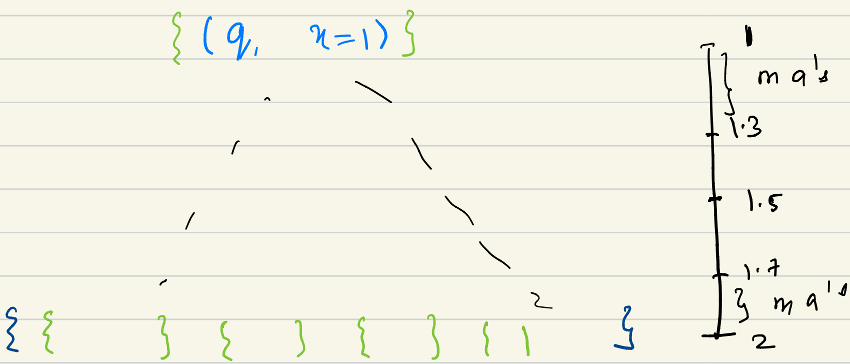
Consider the acceptance game for $\mathcal{A}$ on w_1.

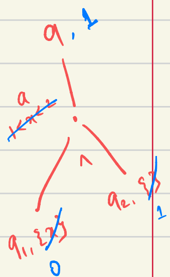
- Let (q, v) be a configuration reached at $t = 1$.

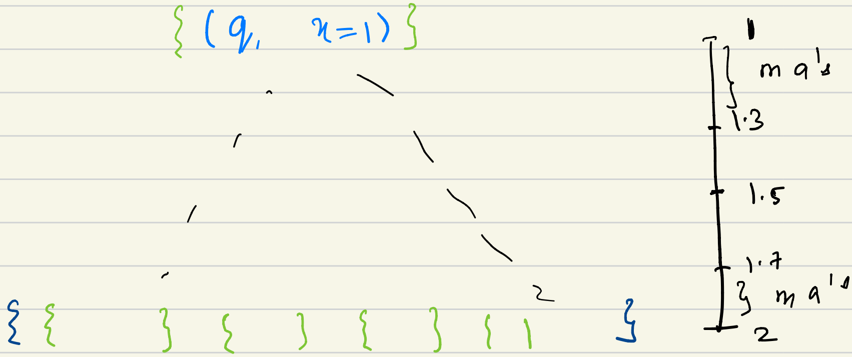
What are the possible values of x at $t = 1$?

$(q_1, x=1)$ $(q_1, x=0.7)$ $(q_1, x=0.3)$

Pick $(q_1, x=1)$ and investigate the set of sets of configns. reached from here after reading the entire word.





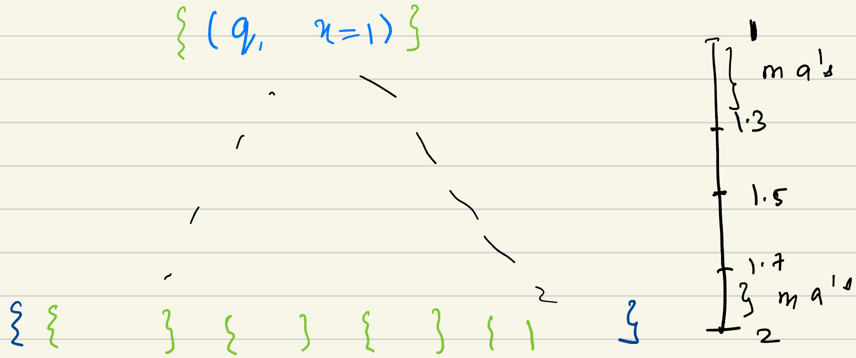


- In $(1, 2)$ transitions with guard $x = 0$ are never used.
- In fact, only those transitions with

  either i) $1 < x < 2$

  or ii) $0 < x < 1$

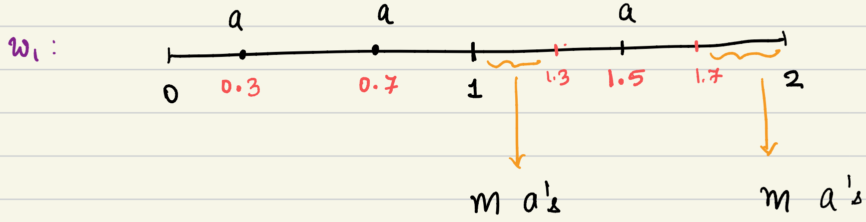
  are used.
- i) is taken until 'x' is reset, (ii) is taken after x is reset.
- Therefore, if we maintain an extra bit 0/1 in each state to mark whether x has been reset until now, we can recover the behaviour of $\mathcal{A}$ in the interval $(1, 2)$.



- Therefore, starting from $(q, x=1)$, the rest of the accepting run is identical to the run of an (untimed) AFA with $2n$ states, starting from $(q, 0)$ to denote not reset.

- From our choice of 'm' and 'l', the same set of sets will be reached by this untimed AFA on the word w_2 !

- Hence, from $(q, x=1)$, w_2 will also be accepted.



(m, l are as chosen before)

Let us now focus on $(q, x = 0.7)$ at $t=1$

- Upto $t=1.3$ the word is the same in both w_1 and w_2 and hence the same set of configuration will be reached at $t=1.3$

- Configurations at $t=1.3$ are either $(q, x=1)$ or $(q, x<0.3)$

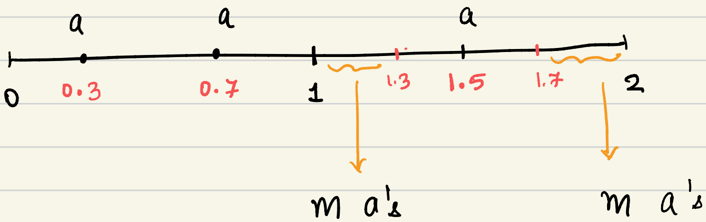
- From $(q, x=1)$, apply same argument as before.

- From $(q, x \leq 0.7)$, only $(0<x<1)$ transitions will be taken, so it behaves like an untimed AFA with 'n' states.

- By our choice of 'm' and 'l' the same set of set of states is reached after reading w_1 and w_2.

Hence from $(q, x=0.7)$ at $t=1$, if w_1 is accepted, w_2 is also accepted.

w_1 :



(m, ℓ are as chosen before)

Finally consider $(q, x = 0.3)$ at $t = 1$.

- upto $t = 1.7$, $\mathcal{A}$ will take only $0 < x < 1$ edges.

- Hence the behaviour is similar to an AFA, and the same set of "states" will be reached for both w_1 and w_2 at $t = 1.7$

  The value of x may be different. However, it will either be $x = 1$ for both words, or some value with $0 < x < 1$ in both.

- . From configurations with $x < 1$ at $t = 1.7$, the actual value remains $0 < x < 1$ for the rest of the word. Hence the true value does not matter.

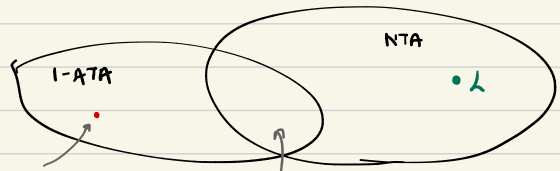
- This shows that the set of set of states reached after both w_1 and w_2 are the same!

If w_1 is accepted by $\mathcal{A}$, w_2 is also accepted by $\mathcal{A}$.

— Contradiction

Summary of Part 1:

Expressive power of 1-ATA Vs many clock NTA



no two a's at distance 1

Contains 1-clock NTAs and some 2-clock NTAs too.

Alternating Timed Automata:

- What we have seen so far?

  - Model is closed under union, intersection, complement

  - Emptiness is undecidable for general ATA

  - Consider 1-clock ATA

    ↳ Expressive power incomparable to many clock NTA.

Today:

- Emptiness is decidable for 1-clock ATA (idea of proof)

- Complexity of the emptiness problem

Algorithm for the emptiness problem for 1-ATA:

Given a 1-clock ATA $\mathcal{A}$, is $\mathcal{L}(\mathcal{A})$ empty?

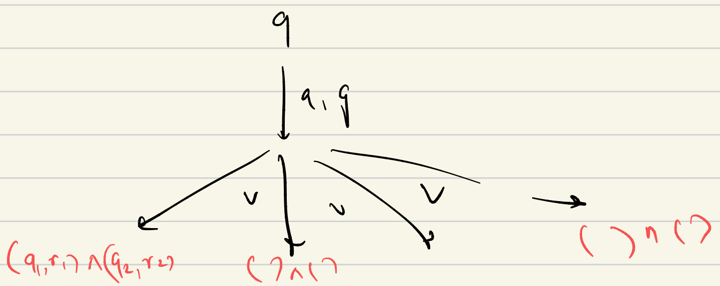
- Algorithm similar to Ouaknine-Worell algorithm for universality of 1-NTA
- Now we need to handle both universal and existential transitions.

Assumption:

- boolean combinations in the transitions are in

disjunctive normal form

$$(\cdot \wedge \cdot \wedge \cdot \cdots) \vee (\cdot \wedge \cdot \wedge \cdot \wedge \cdots) \vee \cdots \vee (\cdot \wedge \cdot \wedge \cdots \wedge)$$



Labelled transition system: $T(A)$

Configuration P: $\{ (q_1, v_1)\ (q_2, v_2), \ldots, (q_k, v_k) \}$

a set of states

(location of automaton, value of clock)

Transitions between configurations:

$$P \xrightarrow{t,a} P'$$

$P = \{(q_1, v_1)\ \ (q_2, v_2)\}$

t, a ... b_1 ... $b_1^1 \vee b_1^2 \vee b_1^3$

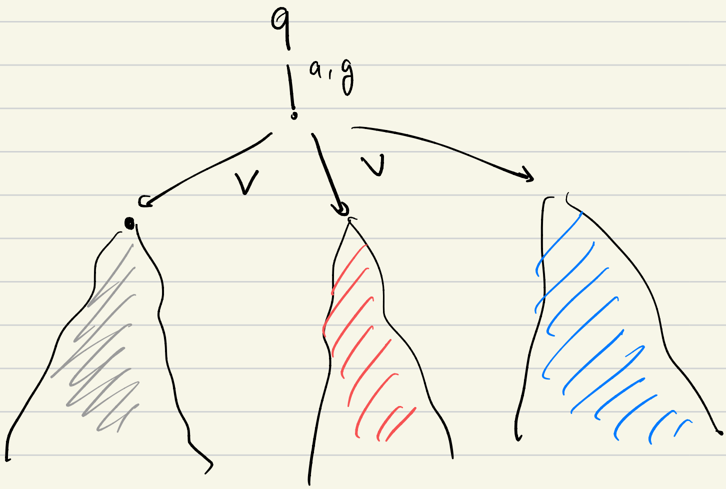
t, a ... b_2 ... $b_2^1 \vee b_2^2$

For each $(q, v) \in P$

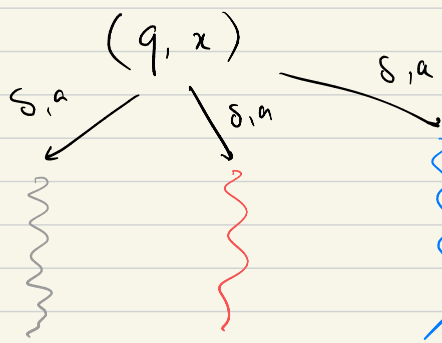
- let $v' = v + t$
- let $b = \delta(q, a, \sigma)$ for the uniquely determined σ satisfied by v'
- Choose one of the disjuncts of b : $(q_1, r_1) \wedge (q_2, r_2) \wedge \ldots \wedge (q_k, r_k)$
- $Next_{(q,v)} := \{ (q_i, v'[r_i := 0]) \mid i = 1, \ldots, k \}$

Then, $P' = \bigcup_{(q,v) \in P} Next_{(q,v)}$

$Next^{b_1^1}_{(q_1 v_1)}$ $\quad Next^{b_1^2}_{(q_1 v_1)}$ $\quad Next^{b_1^3}_{(q_1 v_1)}$ $\quad \Bigg\| \quad$ $Next^{b_2^1}_{(q_2, v_2)}$ $\quad Next^{b_2^2}_{(q_2 v_2)}$



$x + \delta \models g$

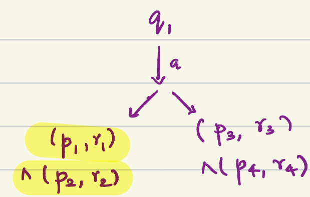


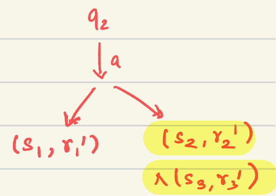
$$\{ (q_1, v_1) \ , \ (q_2, v_2) \}$$

$$\Big\downarrow a, t$$

$$\cdot$$

$$\{ (p_1, \cdot)(p_2, \cdot) \ , \ (s_2, \cdot), (s_3, \cdot) \}.$$





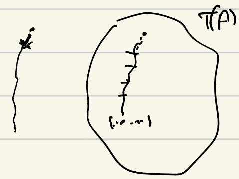
One possible transition in $T(A)$

Good node: All states are accepting

Theorem: $\mathcal{L}(A)$ is non-empty

iff

$T(A)$ has a path to a good node from the initial configuration



Rest of the algorithm similar to OW-05.

Lower bound

Complexity of emptiness of **purely universal** 1-clock ATA is **not** bounded by a **primitive recursive** function

Lower bound

Complexity of emptiness of **purely universal** 1-clock ATA is **not** bounded by a **primitive recursive** function

Emptiness of purely universal 1-ATA $\longrightarrow$ universality of 1-NTA

$A \longrightarrow A^c$ (1-NTA)

$\Rightarrow$ complexity of Ouaknine-Worrell algorithm for **universality** of 1-clock TA is **non-primitive recursive**

Primitive recursive functions

Functions $f : \cancel{\mathbb{N} \mapsto \mathbb{N}}$ $\underline{\underline{\mathbb{N}^k}} \longmapsto \underline{\underline{\mathbb{N}^\ell}}$ $k \geq 0$

Basic primitive recursive functions:

- **Zero function:** $Z() = 0$, Constant function: $C_n^k(x_1, \ldots x_k) = n$
- **Successor function:** $Succ(n) = n + 1$
- **Projection function:** $P_i(x_1, \ldots, x_n) = x_i$

$h \circ (g_1, g_2 \cdots g_m)$

Operations:

$g_1(x_1 \ldots x_k) \quad g_2(x_1 \ldots x_k) \quad \ldots g_m(x_1 \ldots x_k)$

- **Composition** $h(y_1 \quad y_2 \quad y_m)$
- **Primitive recursion:** if f and g are p.r. of arity k and $k+2$, there is a p.r. h of arity $k+1$:

$$
\begin{aligned}
h(0, x_1, \ldots, x_k) &= f(x_1, \ldots, x_k) \\
h(n+1, x_1, \ldots, x_k) &= g(h(n, x_1, \ldots, x_k), n, x_1, \ldots, x_k)
\end{aligned}
$$

$h(n, x_1, \ldots x_k),\ n,\ (x_1, \ldots x_k)$

Composition:

p.r.

$$\begin{cases} g_1 : \mathbb{N}^k \longrightarrow \mathbb{N} & g_1(x_1, \ldots x_k) \longrightarrow y_1 \\ \vdots & \\ g_m : \mathbb{N}^k \longrightarrow \mathbb{N} & g_m(x_1 \ldots x_k) \longrightarrow y_m \\ h : \mathbb{N}^m \longrightarrow \mathbb{N} & \end{cases}$$

$$h \circ (g_1, \ldots g_m)[x_1 \ldots x_k)] \rightarrow h[g_1(x_1 \ldots x_k), g_2(x_1 \ldots x_k), \ldots, g_m(x_1 \ldots x_k)]$$

$\downarrow$

will be p.r. obtained by composition.

Addition:

$$
\begin{aligned}
\overset{h}{Add}(0,y) &= y \qquad f(y)=y \\
h\,Add(n+1,y) &= Succ(Add(n,y))
\end{aligned}
$$

$Succ(P_1(Add(n,y), n, y))$

h: $Succ \circ P_1$

Addition:

$$\begin{aligned} Add(0, y) &= y \\ Add(n+1, y) &= Succ(Add(n, y)) \end{aligned}$$

$Succ(P_1(Add(n,y), n, y))$

Multiplication:

$$\begin{aligned} Mult(0, y) &= Z() \\ Mult(n+1, y) &= Add(Mult(n, y), y) \end{aligned}$$

$P_1(Mult(n,y), n, y) = Mult(n,y)$

$P_3(Mult(n,y), n, y) = y$

$Add \circ (P_1, P_3) : (Mult(n,y), n, y)$

$= Add(P_1(\ ), P_3(\ ))$

$= Add(Mult(n,y), y)$

Addition:

$$
\begin{aligned}
Add(0, y) &= y \\
Add(n+1, y) &= Succ(Add(n, y))
\end{aligned}
$$

Multiplication:

$$
\begin{aligned}
Mult(0, y) &= Z() \\
Mult(n+1, y) &= Add(Mult(n, y), y)
\end{aligned}
$$

Exponentiation 2^n:

$$
\begin{aligned}
Exp(0) &= Succ(Z()) \\
Exp(n+1) &= Mult(Exp(n), 2)
\end{aligned}
$$

$P_1[Exp(n), n]$

C_2^2

$Mult \circ (P_1, C_2^2)$

Addition:

$$\begin{aligned} Add(0, y) &= y \\ Add(n+1, y) &= Succ(Add(n, y)) \end{aligned}$$

Multiplication:

$$\begin{aligned} Mult(0, y) &= Z() \\ Mult(n+1, y) &= Add(Mult(n, y), y) \end{aligned}$$

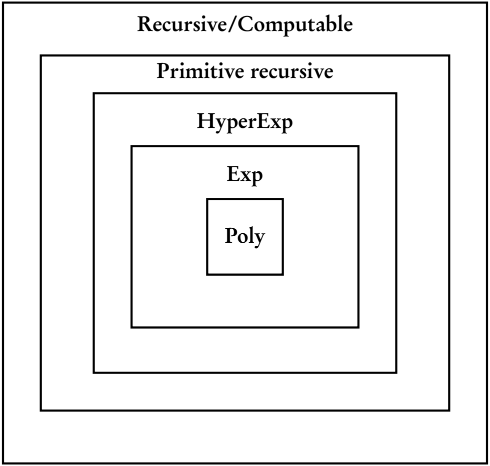
Exponentiation 2^n:

$$\begin{aligned} Exp(0) &= Succ(Z()) \\ Exp(n+1) &= Mult(Exp(n), 2) \end{aligned}$$

HyperExp(1) = 2
HyperExp(2) = 2^2
(3) = 2^{2^2}
(4) = $2^{2^{2^2}}$

Hyper-exponentiation (tower of n two-s):

$$\begin{aligned} HyperExp(0) &= Succ(Z()) \\ HyperExp(n+1) &= Exp(HyperExp(n)) \end{aligned}$$



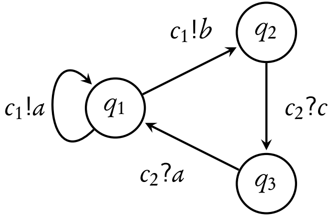
Recursive but not primitive rec.: Ackermann function, Sudan function

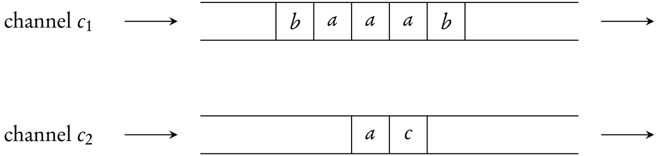
Coming next: a problem that has complexity non-primitive recursive

Channel systems

$$(q, w) \xrightarrow{c!a} (q', aw)$$

$$(q, wa) \xrightarrow{c?a} (q', w)$$





Finite state description of communication protocols
G. von Bochmann. 1978

On communicating finite-state machines
D. Brand and P. Zafiropulo. 1983

Theorem [BZ'83]

Reachability in channel systems is **undecidable**

Coming next: modifying the model for decidability

Lossy channel systems

Finkel'94, Abdulla and Jonsson'96

Messages stored in channel can be **lost** during transition

$(q, w) \xrightarrow{c!a} (q', w')$ where w' is a subword of aw

$(q, wa) \xrightarrow{c?a} (q', w'')$ where w'' is a subword of w

Lossy channel systems

Finkel'94, Abdulla and Jonsson'96

Messages stored in channel can be **lost** during transition

Theorem [Schnoebelen'2002]

Reachability for **lossy one-channel** systems is **non-primitive recursive**

Reachability problem for **lossy one-channel** systems can be reduced to emptiness problem for **purely universal 1-clock ATA**

1-clock ATA

- **closed** under boolean operations
- **decidable** emptiness problem
- expressivity **incomparable** to many clock TA
- **non-primitive recursive** complexity for emptiness

1-clock ATA

- **closed** under boolean operations
- **decidable** emptiness problem
- expressivity **incomparable** to many clock TA
- **non-primitive recursive** complexity for emptiness

- Other results: **Undecidability** of:
  - 1-clock ATA + ε-transitions
  - 1-clock ATA over infinite words