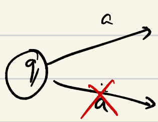
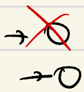
DETERMINISTIC TIMED AUTOMATA

- 1. Definition and examples
- 2. Closure properties
- 3. Decision problems

1. Definition and examples

In untimed setting:



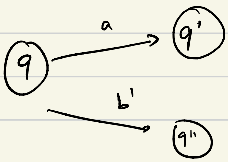


Determinism: i) at most one transition on 'a' at every state.

ii) unique initial state

i) and ii) will ensure that there is at most one run on every word

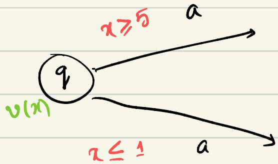
Completeness:



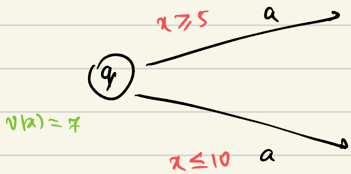
There is a transition from every state on every letter.

" There is exactly one run on every word when the automaton is deterministic and complete. "

In the timed setting:



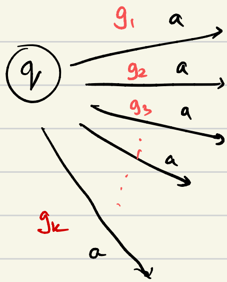
Any valuation v can take only one of the two transitions.



Not allowed.

Deterministic timed automaton:

It is a TA with a single initial state and with an additional condition on transition syntax:



$g_i \cap g_j = \phi$ for every pair i,j

Let q be an arbitrary state and let $g_1, g_2, \ldots g_k$ be the guards on outgoing transitions from 'q' on a letter 'a': then $g_i \wedge g_j = \phi$ $\forall i,j$.

For every pair of transitions (q, g, a, R, q_1) and (q, g', a, R', q_2) we have $g \wedge g'$ is unsatisfiable

A TA is said to be complete if for every state 'q' and every letter 'a':

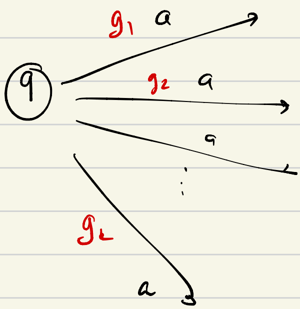
if the set of transitions on 'a' from 'q' are

(q, g_1, a, R_1, q_1) (q, g_2, a, R_2, q_2) .... (q, g_k, a, R_k, q_k)

then $g_1 \cup g_2 \cup g_3 \ldots \cup g_k$ = set of all valuations

considering guards as sets of valuation

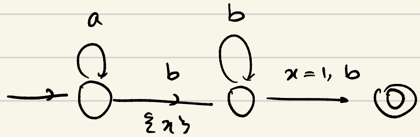
$g_1 \vee g_2 \vee g_3 \vee \ldots \vee g_k$ is the "true" constraint



$g_1 \vee g_2 \ldots \vee g_k$ = true

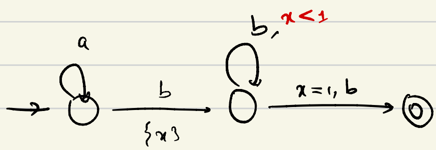
Example 1:

$\mathcal{A}$:



Non- deterministic

$\mathcal{B}$:

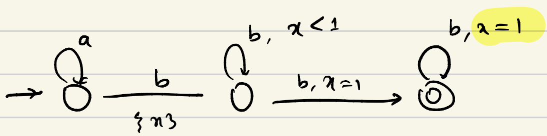


deterministic.

↳ Is this language equivalent?

w: $(b, 0)$ $(b, 1)$, $(b, 1)$

$w \in \mathcal{L}(\mathcal{A})$, but $w \notin \mathcal{L}(\mathcal{B})$

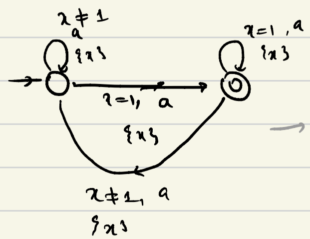


Example 2:

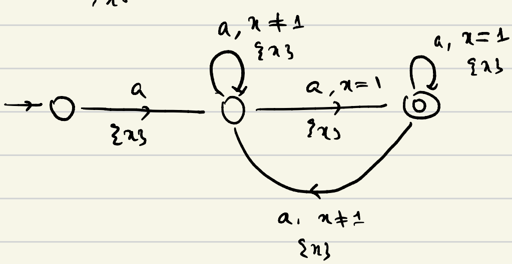
Words over unary alphabet $\{a\}$

$(\sigma_1 \sigma_2 \dots \sigma_k, \; \tau_1 \tau_2 \dots \tau_k)$ s.t. $k \geq 2$ and $\tau_k - \tau_{k-1} = 1$

- distance between last two letters is 1.



→ does not take care of $k \geq 2$



CLOSURE PROPERTIES OF DTA :

- Union: DTAs $\mathcal{A}_1$ and $\mathcal{A}_2$ Assume both are complete.

Product construction.

$\mathcal{A}_1 = (Q_1, \Sigma_1, X_1, \Delta_1, F_1)$

$\mathcal{A}_2 = (Q_2, \Sigma_1, X_2, \Delta_2, F_2)$

We assume $Q_1 \cap Q_2 = \phi$

$X_1 \cap X_2 = \phi$

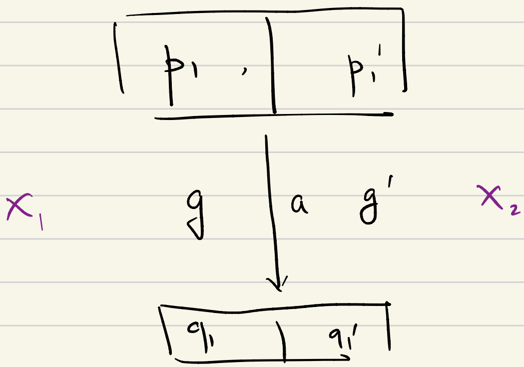
$\mathcal{A}_{union}$: $(Q_1 \times Q_2, \Sigma_1, X_1 \cup X_2, \Delta, Q_1 \times F_2 \cup F_1 \times Q_2)$

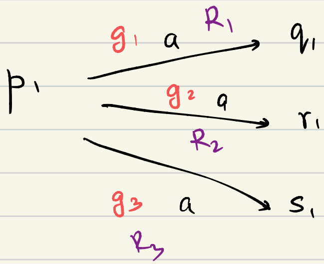
$$(p_1, p_1') \xrightarrow[R \cup R']{a,\ g \wedge g'} (p_2, p_2')$$

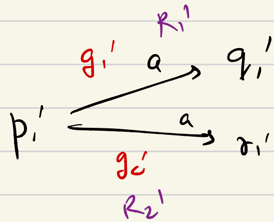
for every pair of transitions:

$(p_1, a, g, R, p_2) \in \Delta_1$ and

$(p_1', a, g, R, p_2) \in \Delta_2$







$$\left.\begin{array}{l} g_1 \wedge g_2 \\ g_2 \wedge g_3 \\ g_3 \wedge g_1 \end{array}\right\} \text{unsatisfiable}$$

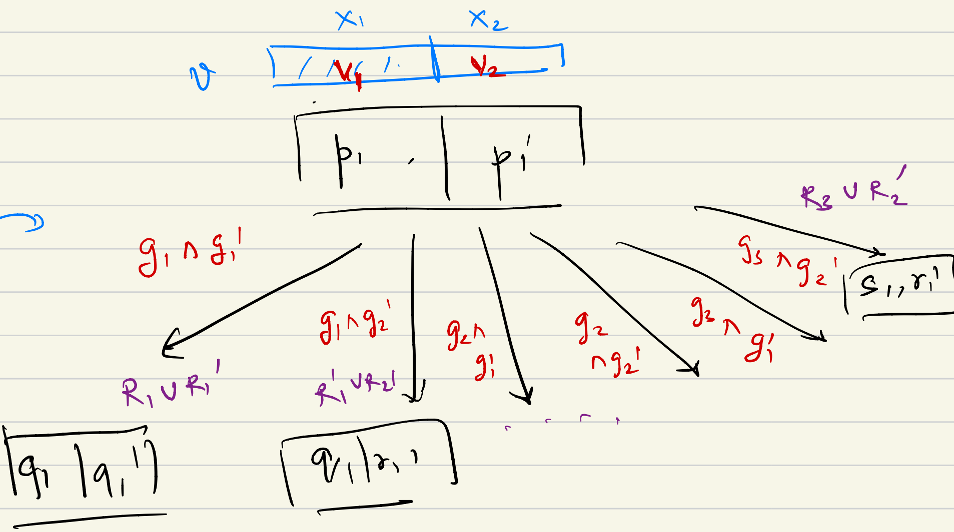
$g_1' \wedge g_2'$ is unsat.

$g_1' \vee g_2'$ is T.

$g_1 \vee g_2 \vee g_3$ is "true"

T.

pairwise disjoint



Thm: A_{union} is deterministic, complete and accepts $\mathcal{L}(A_1) \cup \mathcal{L}(A_2)$.

- <u>Intersection:</u> $\mathcal{A}_1, \mathcal{A}_2$: complete and deterministic

$\mathcal{A}_{intersection}$: $(Q_1 \times Q_2, \Sigma, X_1 \cup X_2, \Delta, F_1 \times F_2)$

same as in $\mathcal{A}_{union}$.

<u>Exercise:</u> Do these constructions work for complete non-deterministic T.A ?

- <u>Complementation:</u> $\mathcal{A}$: deterministic T.A

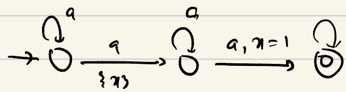
  - Assume $\mathcal{A}$ is complete

    ↳ For every timed word there is a unique run

So complement language is accepted by swapping the accept and non-accept states

Thm: Non-deterministic T.A are strictly more expressive than deterministic T.A.

Proof. Consider: $\mathcal{A}$:



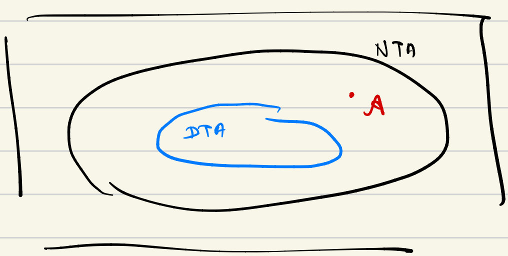
there exist 2 a's which are at distance 1 apart.

We have seen earlier that $\mathcal{L}(\mathcal{A})^c$ is not even timed regular

– There is no T.A. accepting $\mathcal{L}(\mathcal{A})^c$.

If $\mathcal{L}(\mathcal{A})$ has a DTA, then $\mathcal{L}(\mathcal{A})^c$ has a DTA – contradiction.

$\Rightarrow$ $\mathcal{L}(\mathcal{A})$ cannot be recognized by a deterministic T.A.



Decision problems:

- 1. Given DTA $\mathcal{A}$, is $\mathcal{L}(\mathcal{A})$ empty?

    → Build region automaton.

- 2. Given DTA $\mathcal{A}$, is $\mathcal{L}(A)$ universal? $\mathcal{L}(\mathcal{A}) = T\Sigma^*$?

    → Decidable

    → Complement and check for emptiness.

- 3. Given an NTA $\mathcal{A}$, does there exist a DTA $\mathcal{B}$ s.t. $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$?

    → undecidable (Finkel '06)

Summary:

- Deterministic TA., closure properties, decision problems