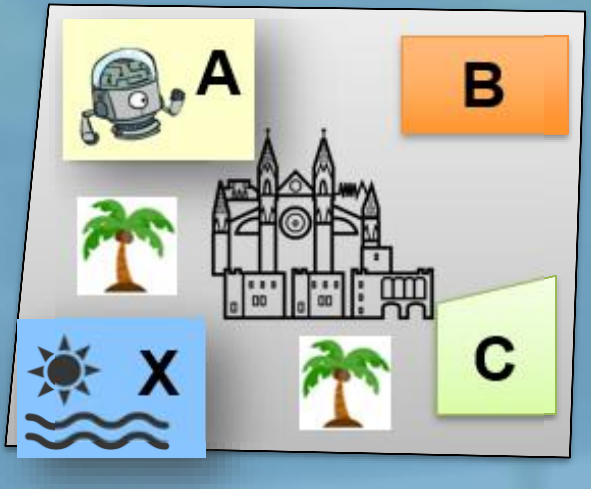
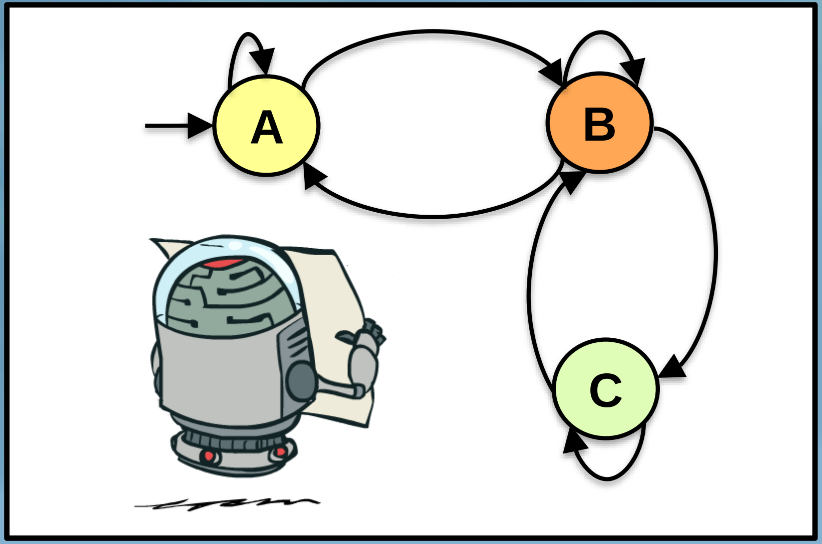
Graz University of Technology
Institute for Applied Information Processing and Communications

Automata and LTL Model Checking

Bettina Könighofer





Model Checking SS23

May 11th 2023

Model Checking of LTL

given an LTL property φ and a Kripke structure M
check whether $M \models \varphi$

1. Construct $\neg\varphi$
2. Construct a **Büchi** automaton $\mathcal{S}_{\neg\varphi}$
3. **Translate M to an automaton $\mathcal{A}$.**
4. Construct the automaton $\mathcal{B}$ with $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{S}_{\neg\varphi})$
5. If $\mathcal{L}(\mathcal{B}) = \emptyset \Rightarrow \mathcal{A}$ satisfies φ
6. Otherwise, a word $v \cdot w^{\omega} \in \mathcal{L}(\mathcal{B})$ is a counterexample
   - a computation in M that does not satisfy φ

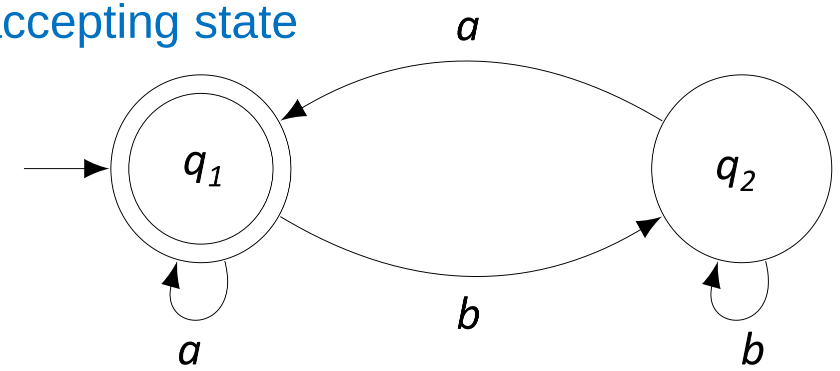
Outline

- Finite automata on finite words
- Automata on infinite words (Büchi automata)
- Deterministic vs non-deterministic Büchi automata
- Intersection of Büchi automata
- Checking emptiness of Büchi automata
- Generalized Büchi automata
- **Model checking using automata**
- Translation of LTL to Büchi automata

Finite Automata on **Finite** Words

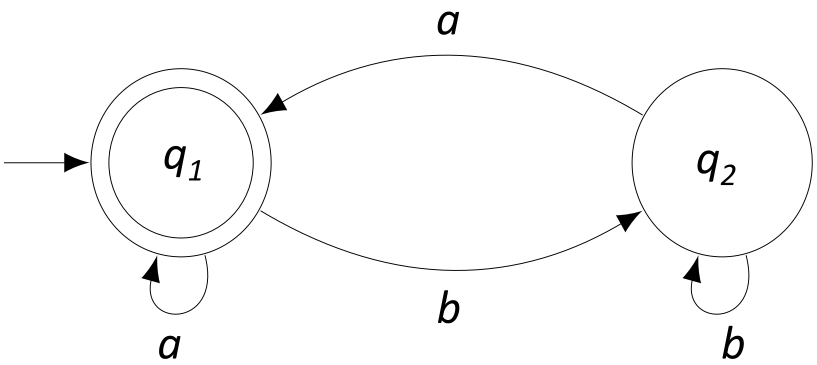
Regular Automata

- $\mathcal{A} = (\Sigma, Q, \Delta, Q^0, F)$
- Σ is the finite alphabet
- Q is the finite set of states
- $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation
- Q^0 is the set of initial states
- F is the set of accepting states
  - $\mathcal{A}$ accepts a word if there is a corresponding run ending in an accepting state



Finite Automata on Finite Words
Regular Automata

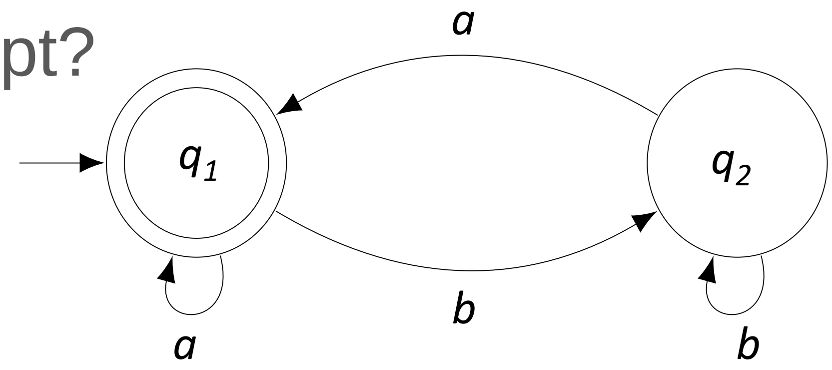
- Example: $\mathcal{A} = (\mathbf{\Sigma}, \mathbf{Q}, \mathbf{\Delta}, \mathbf{Q}^0, \mathbf{F})$
- $\mathbf{\Sigma} = \{a, b\}$
- $\mathbf{Q} = \{q_1, q_2\}$
- $\mathbf{\Delta} = \{ (q_1, a, q_1), (q_1, b, q_2), (q_2, a, q_1), (q_2, b, q_2)\},$
- $\mathbf{Q}^0 = \{q_1\}$
- $\mathbf{F} = \{q_1\}$



Finite Automata on Finite Words
Regular Automata

- Example: $\mathcal{A} = (\mathbf{\Sigma}, \mathbf{Q}, \mathbf{\Delta}, \mathbf{Q}^0, \mathbf{F})$
- $\mathbf{\Sigma} = \{a, b\}$
- $\mathbf{Q} = \{q_1, q_2\}$
- $\mathbf{\Delta} = \{ (q_1, a, q_1), (q_1, b, q_2), (q_2, a, q_1), (q_2, b, q_2)\},$
- $\mathbf{Q}^0 = \{q_1\}$
- $\mathbf{F} = \{q_1\}$

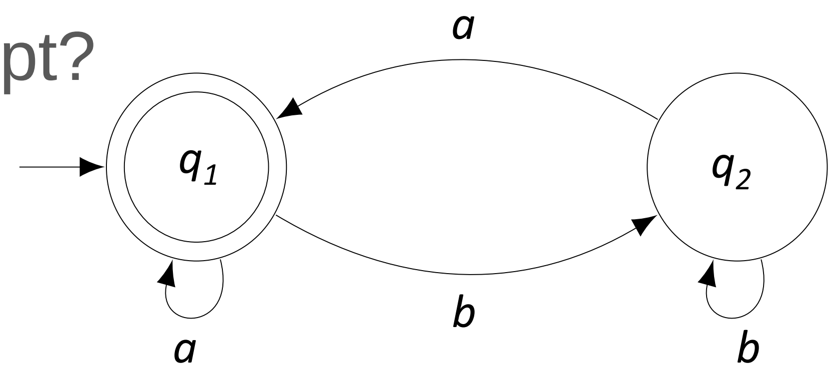
ToDo What words does it accept?



Finite Automata on Finite Words

Regular Automata

- Example: $\mathcal{A} = (\mathbf{\Sigma},\mathbf{Q},\mathbf{\Delta},\mathbf{Q}^0,\mathbf{F})$
- $\mathbf{\Sigma} = \{a, b\}$
- $\mathbf{Q} = \{q_1, q_2\}$
- $\mathbf{\Delta} = \{ (q_1, a, q_1), (q_1, b, q_2), (q_2, a, q_1), (q_2, b, q_2)\},$
- $\mathbf{Q}^0 = \{q_1\}$
- $\mathbf{F} = \{q_1\}$
- What words does it accept?



$\mathcal{L}(\mathcal{A})$ = {the empty word} ∪ {all words that end with a}

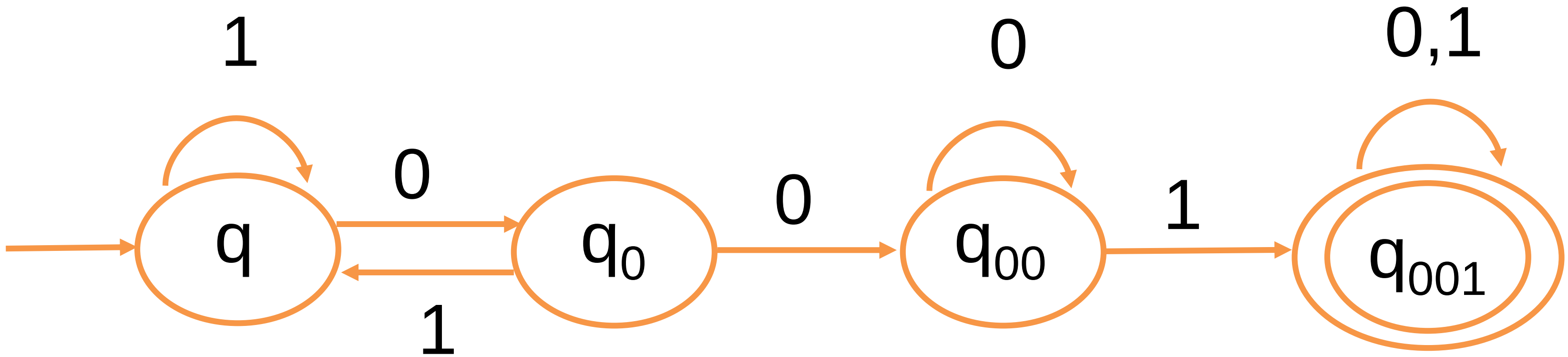
$= \{\varepsilon\} \cup \{a,b\}^*a$

Finite Automata on Finite Words
Regular Automata

ToDo: Build an automaton that accepts all and only those strings that contain 001

Finite Automata on Finite Words
Regular Automata

Build an automaton that accepts all and only those strings that contain 001



Languages on Finite Automata

- Given a word $v=a_1,a_2,\ldots,a_n$ and automaton $\mathcal{A}$
- A run $\boldsymbol{\rho} = q_0,q_1,\ldots q_n$ of $\mathcal{A}$ over v is a sequence of states s.t.
  - $q_0 \in \mathbf{Q}^0$
  - for all $0 \leq i \leq n\text{-}1$, $(q_i,a_{i+1},q_{i+1}) \in \boldsymbol{\Delta}$
  - → $\boldsymbol{\rho}$ is a path in the graph of $\mathcal{A}$.

Languages on Finite Automata

- Given a word $v=a_1,a_2,\ldots,a_n$ and automaton $\mathcal{A}$
- A run $\boldsymbol{\rho} = q_0,q_1,\ldots q_n$ of $\mathcal{A}$ over v is a sequence of states s.t.
  - $q_0 \in \mathbf{Q}^0$
  - for all $0 \leq i \leq n-1$, $(q_i,a_{i+1},q_{i+1}) \in \boldsymbol{\Delta}$
  - → $\boldsymbol{\rho}$ is a path in the graph of $\mathcal{A}$.
- A run is accepting ⇔

Languages on Finite Automata

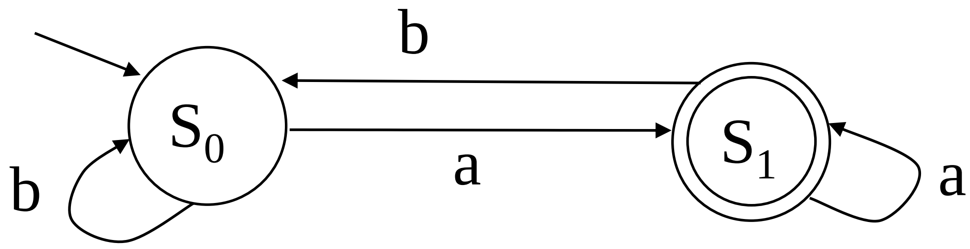
- Given a word $v=a_1,a_2,\ldots,a_n$ and automaton $\mathcal{A}$
- A run $\boldsymbol{\rho} = q_0,q_1,\ldots q_n$ of $\mathcal{A}$ over v is a sequence of states s.t.
  - $q_0 \in \mathbf{Q}^0$
  - for all $0 \leq i \leq n-1$, $(q_i,a_{i+1},q_{i+1}) \in \boldsymbol{\Delta}$
  - → $\boldsymbol{\rho}$ is a path in the graph of $\mathcal{A}$.
- A run is accepting ⇔ $q_n \in \mathbf{F}$

Languages on Finite Automata

- Given a word $v=a_1,a_2,\ldots,a_n$ and automaton $\mathcal{A}$
- A run $\boldsymbol{\rho} = q_0,q_1,\ldots q_n$ of $\mathcal{A}$ over v is a sequence of states s.t.
  - $q_0 \in \mathbf{Q}^0$
  - for all $0 \leq i \leq n-1$, $(q_i,a_{i+1},q_{i+1}) \in \boldsymbol{\Delta}$
  - → $\boldsymbol{\rho}$ is a path in the graph of $\mathcal{A}$.
- A run is accepting ⇔ $q_n \in \mathbf{F}$
- Language of $\mathcal{A}$
  - $\mathcal{L}(\mathcal{A}) \subseteq \boldsymbol{\Sigma}^*$, is the set of words that $\mathcal{A}$ accepts.
- Languages accepted by finite automata are ***regular languages***.

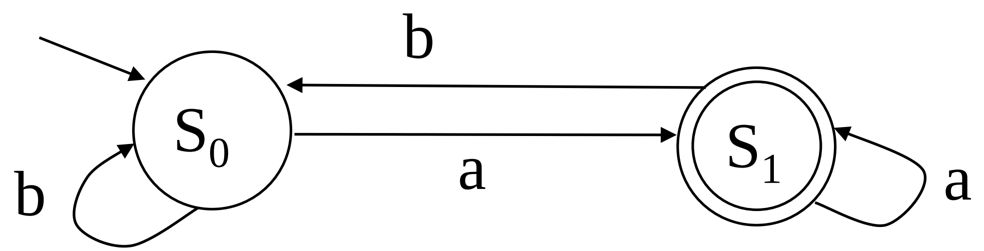
Deterministic & Non-Deterministic Automata

- $\mathcal{A}$ is deterministic if $\mathbf{\Delta}$ is a function (one output for each input).
  - $|\mathbf{Q}^0| = 1$, and
  - $\forall q \in \mathbf{Q}\ \forall a \in \mathbf{\Sigma}\mathbf{:}\ |\ \mathbf{\Delta}(q,a)\ | \leq 1$
- Det. automata have exactly one run for each word.

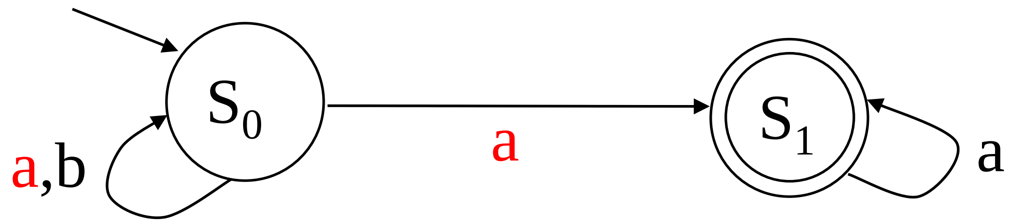


Deterministic & Non-Deterministic Automata

- $\mathcal{A}$ is deterministic if **Δ** is a function (one output for each input).
  - $|\mathbf{Q}^0| = 1$, and
  - $\forall q \in \mathbf{Q}\ \forall a \in \mathbf{\Sigma}\mathbf{:}\ |\ \mathbf{\Delta}(q,a)\ | \leq 1$
- Det. automata have exactly one run for each word.



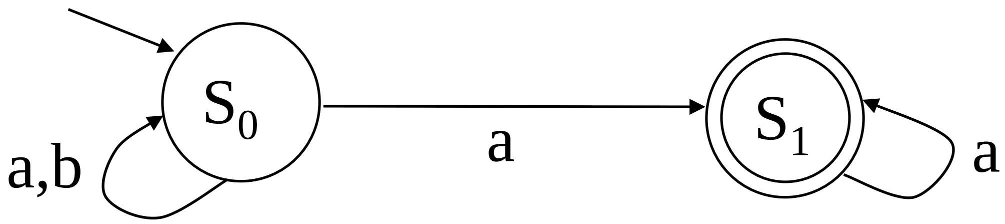
- Non-det. automata
  - Can have ε-transitions (transitions without a letter)
  - Can have transitions $(q,a,q'),(q,a,q'') \in \mathbf{\Delta}$ and $q'' \neq q'$



Nondeterministic Finite Automata (NFA)

- NFA accepts all words that have a run that ends in an accepting state

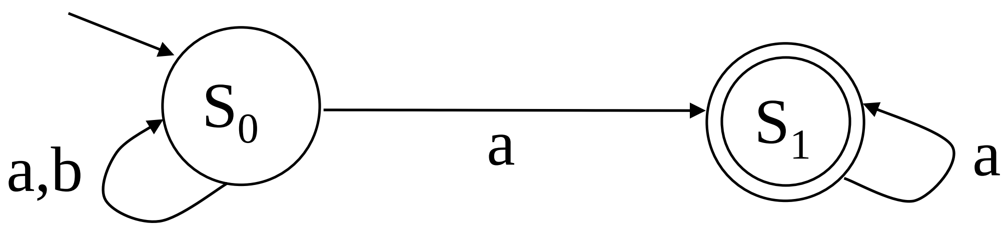
ToDo What is the language of this automaton?



Nondeterministic Finite Automata (NFA)

- NFA accepts all words that have a run that ends in an accepting state
- What is the language of this automaton?

$\mathcal{L}(\mathcal{A})$ = {all words that end with a}



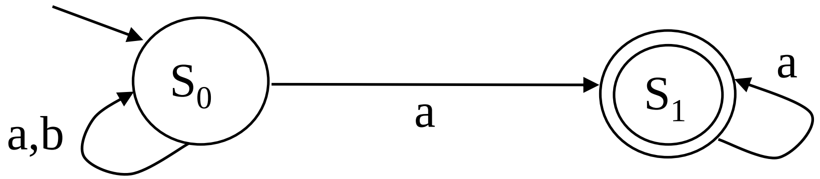
Equivalent deterministic automaton

- Every NFA can be transformed to DFA.
- **Subset-Construction (exponential blow-up)**
  - NFA: $\mathcal{A} = (\mathbf{\Sigma},\mathbf{Q},\mathbf{\Delta},\mathbf{Q}^0,\mathbf{F})$
  - DFA: $\mathcal{A}' = (\mathbf{\Sigma}, P(\mathbf{Q}),\mathbf{\Delta}',\{\mathbf{Q}^0\},\mathbf{F}')$ such that
    - $\mathbf{\Delta}': P(Q) \times \Sigma \rightarrow P(Q)$ where $(Q_1, a, Q_2) \in \Delta'$ if

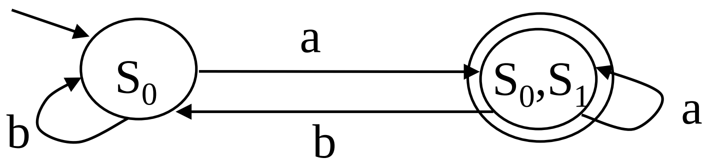
$$Q_2 = \bigcup_{q \in Q_1} \{q' \mid (q, a, q') \in \Delta\}$$

    - $F' = \{Q' \mid Q' \cap F \neq \emptyset\}$

Non-deterministic automaton $\mathcal{A}$



Equivalent Det. automaton $\mathcal{A}'$



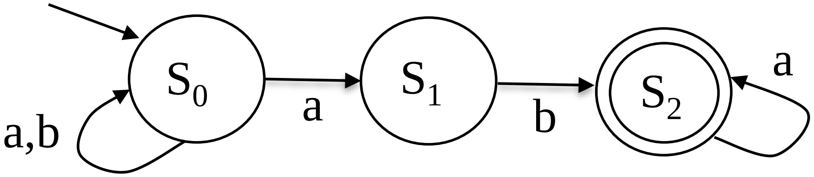
Equivalent deterministic automaton

- Compute the equivalent DFA
  - $\mathcal{A}'$ = ($\mathbf{\Sigma}$, P($\mathbf{Q}$),$\mathbf{\Delta}'$,{$\mathbf{Q}^0$},$\mathbf{F}'$) such that
    - $\mathbf{\Delta}'$: $P(Q) \times \Sigma \rightarrow P(Q)$ where $(Q_1, a, Q_2) \in \Delta'$ if

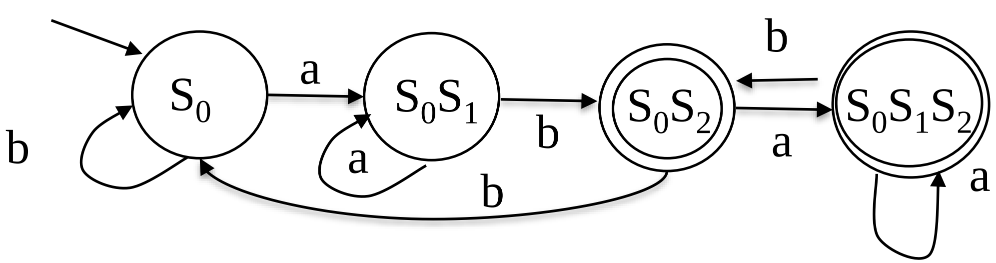
$$Q_2 = \bigcup_{q \in Q_1} \{q' | (q, a, q') \in \Delta\}$$

    - $F' = \{Q' | Q' \cap F \neq \emptyset\}$

Non-deterministic automaton $\mathcal{A}$



Equivalent Det. automaton $\mathcal{A}'$



Equivalent deterministic automaton

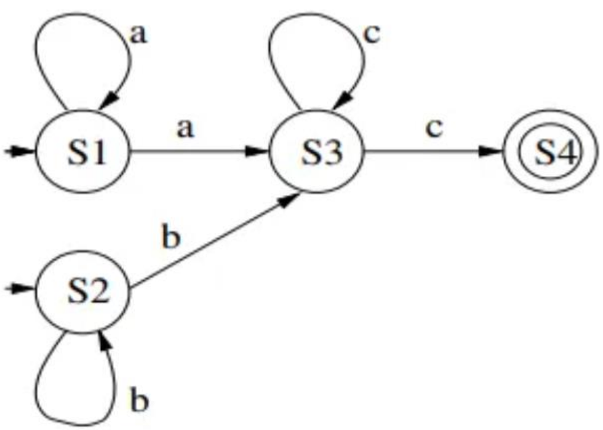
- Compute the equivalent DFA
  - $\mathcal{A}'$ = ($\mathbf{\Sigma}$, P($\mathbf{Q}$),$\mathbf{\Delta}'$,{$\mathbf{Q}^0$},$\mathbf{F}'$) such that
    - $\mathbf{\Delta}'$: $P(Q) \times \Sigma \rightarrow P(Q)$ where $(Q_1, a\ , Q_2) \in \Delta'$ if

$$Q_2 = \bigcup_{q \in Q_1} \{q' | (q, a, q') \in \Delta\}$$

    - $F' = \{Q' | Q' \cap F \neq \emptyset\}$

Non-deterministic automaton $\mathcal{A}$

Equivalent Det. automaton $\mathcal{A}'$



Secure & Correct Systems

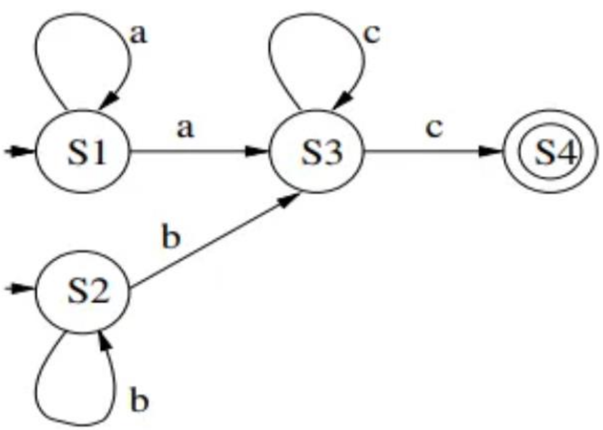
Equivalent deterministic automaton

- Compute the equivalent DFA
  - $\mathcal{A}' = (\Sigma, P(Q), \Delta', \{Q^0\}, F')$ such that
    - Δ': $P(Q) \times \Sigma \rightarrow P(Q)$ where $(Q_1, a, Q_2) \in \Delta'$ if

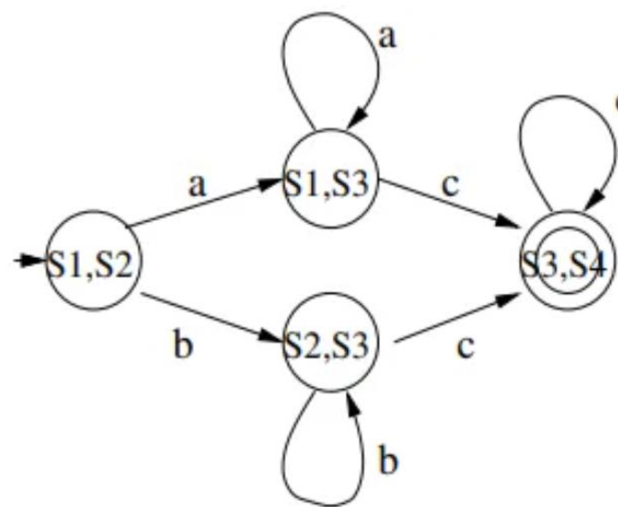
$$Q_2 = \bigcup_{q \in Q_1} \{q' \mid (q, a, q') \in \Delta\}$$

    - $F' = \{Q' \mid Q' \cap F \neq \emptyset\}$

Non-deterministic automaton $\mathcal{A}$



Equivalent Det. automaton $\mathcal{A}'$



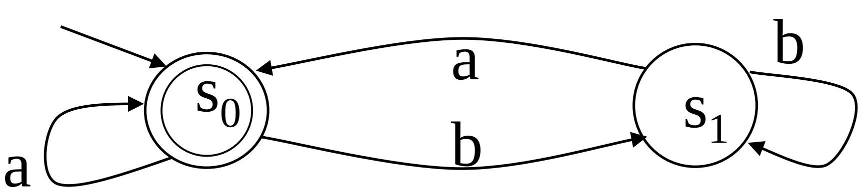
SCOS
Secure & Correct Systems

Complement of DFA

- The complement automaton $\overline{\mathcal{A}}$ accepts exactly those words that are rejected by $\mathcal{A}$

How do we construct $\overline{\mathcal{A}}$?

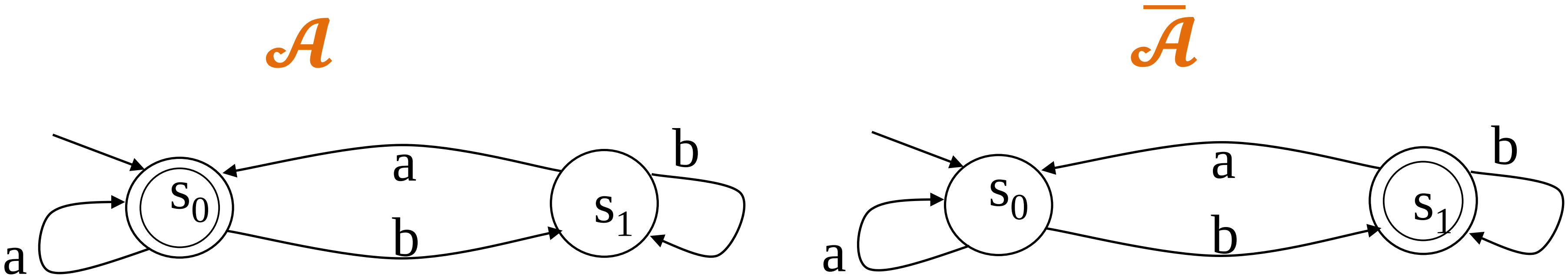
$\mathcal{A}$



$\overline{\mathcal{A}}$=?

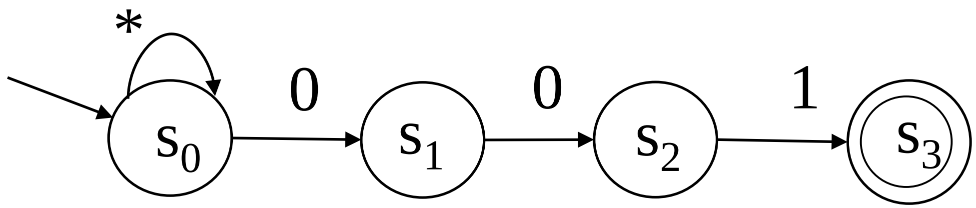
Complement of DFA

- The complement automaton $\bar{\mathcal{A}}$ accepts exactly those words that are rejected by $\mathcal{A}$
- Construction of $\bar{\mathcal{A}}$
  - Substitution of accepting and non-accepting states



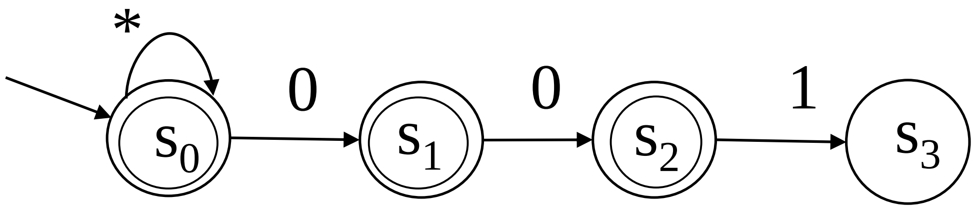
Consider NFA that accepts words that end with 001

$\mathcal{A}$



Let's try switching accepting and non-accepting states:

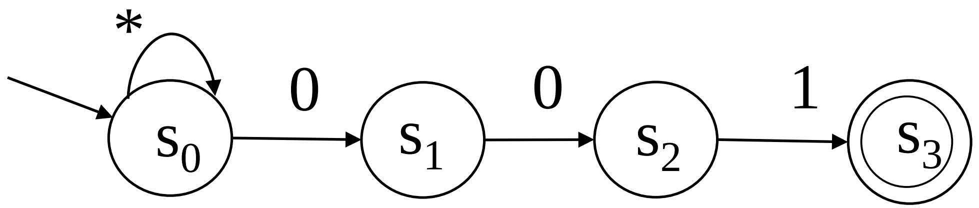
$\bar{\mathcal{A}}$



ToDo *Is* $\bar{\mathcal{A}}$ the complement of $\mathcal{A}$?

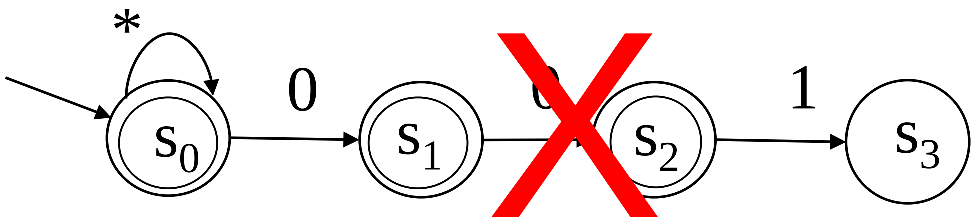
Consider NFA that accepts words that end with 001

$\mathcal{A}$



Let's try switching accepting and non-accepting states:

$\bar{\mathcal{A}}$



The language of this automaton is {0,1}* - this is wrong!

Complement of NFA

- The complement automaton $\overline{\mathcal{A}}$ accepts exactly those words that are rejected by $\mathcal{A}$
- Construction of $\overline{\mathcal{A}}$

Complement of NFA

- The complement automaton $\overline{\mathcal{A}}$ accepts exactly those words that are rejected by $\mathcal{A}$
- Construction of $\overline{\mathcal{A}}$
  1. Determinization: Convert NFA to DFA
  2. Substitution of accepting and non-accepting states

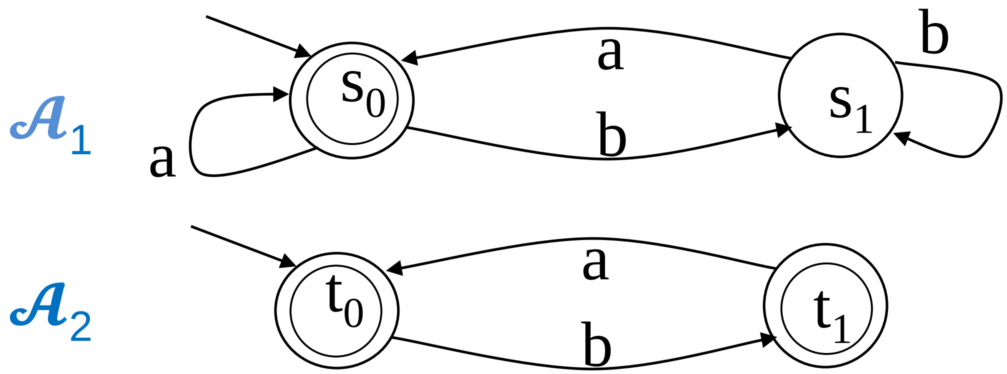
Intersections of NFAs

- Given two languages, L_1 and L_2, the intersection of L_1 and L_2 is

$$L_1 \cap L_2 = \{\, w \mid w \in L_1 \text{ and } w \in L_2 \,\}$$

- Product automaton of $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$ has $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$

Intersections of NFAs



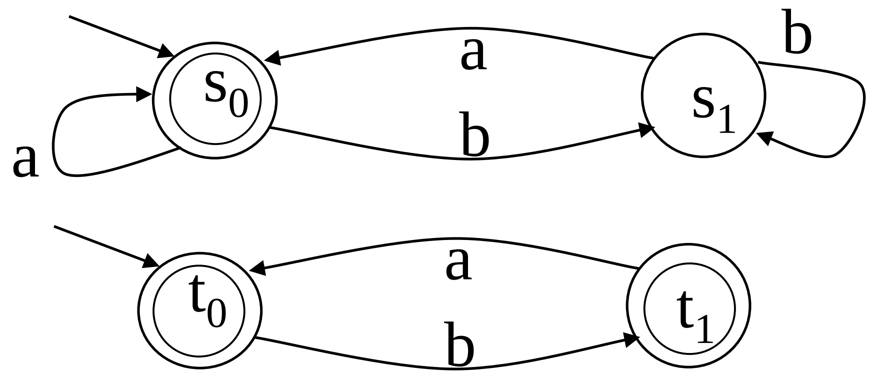
$\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$

1. States: (s_0,t_0), (s_0,t_1), (s_1,t_0), (s_1,t_1).
2. Initial state: (s_0,t_0).
3. Accepting states: (s_0,t_0), (s_0,t_1).

Intersections of NFAs

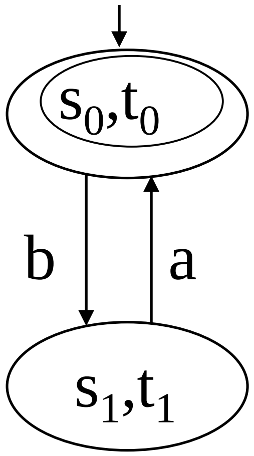
$\mathcal{A}_1$

$\mathcal{A}_2$



$\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$

1. States: (s_0,t_0), (s_0,t_1), (s_1,t_0), (s_1,t_1).
2. Initial state: (s_0,t_0).
3. Accepting states: (s_0,t_0), (s_0,t_1).

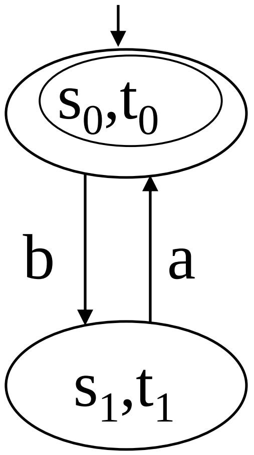


Intersections of NFAs

- Given two languages, L_1 and L_2, the intersection of L_1 and L_2 is

  $$L_1 \cap L_2 = \{ w \mid w \in L_1 \text{ and } w \in L_2 \}$$

- Product automaton of $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$ has $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$
  - $Q = Q_1 \times Q_2$ (Cartesian product),
  - $\Delta\big((q_1, q_2), a\big) = (\Delta_1(q_1, a), \Delta_2(q_2, a))$
  - $q_0 = (q_{01}, q_{02})$
  - $(q_1, q_2) \in F$ *iff* $q_1 \in F_1$ *and* $q_2 \in F_2$



Outline

- Finite automata on finite words
- Automata on infinite words (Büchi automata)
- Deterministic vs non-deterministic Büchi automata
- Intersection of Büchi automata
- Checking emptiness of Büchi automata
- Generalized Büchi automata
- Model checking using automata

Automata on **Infinite** Words (Büchi)

$$\mathcal{B} = (\Sigma, Q, \Delta, Q^0, F)$$

- An infinite run ρ is accepting $\Leftrightarrow$ it visits an accepting state an infinite number of times.
  - $\mathrm{inf}(\rho) \cap F \neq \emptyset$
- $\mathcal{L}(\mathcal{B}) \subseteq \Sigma^\omega$ is the set of all infinite words that $\mathcal{B}$ accepts
- Languages accepted by finite automata on infinite words are called ***ω-regular languages*.**

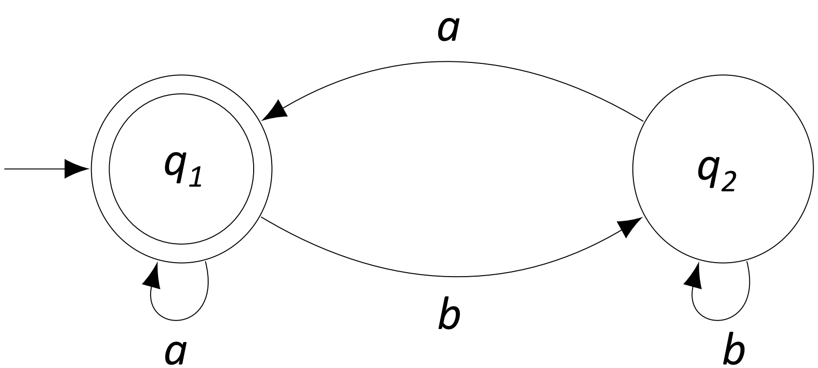
Automata on **Infinite** Words (Büchi)

$$\mathcal{B} = (\Sigma, Q, \Delta, Q^0, F)$$

- ρ is accepting $\Leftrightarrow$ $\mathbf{inf(\rho)} \cap \mathbf{F} \neq \emptyset$

ToDo

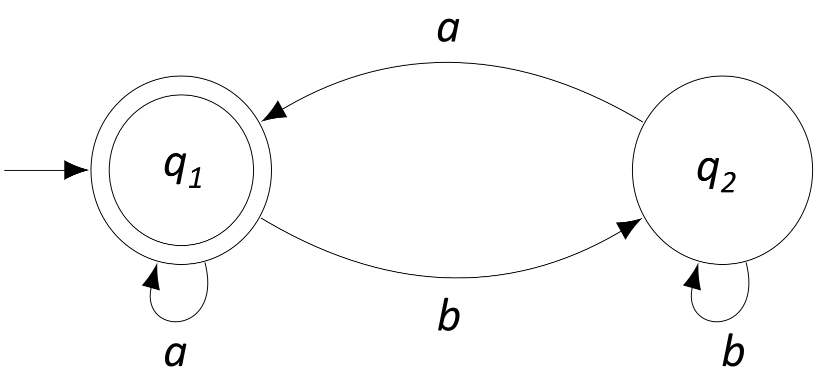
- What is the language of this automaton?



Automata on Infinite Words (Büchi)

$\mathcal{B} = (\Sigma, Q, \Delta, Q^0, F)$

- ρ is accepting $\Leftrightarrow$ $\inf(\rho) \cap F \neq \emptyset$
- Language of Büchi Automaton $\mathcal{B}$



$\mathcal{L}(\mathcal{B})$ = {words with an Infinite number of a's}

or

$\mathcal{L}(\mathcal{B}) = (\{a,b\}^{*}a)^{\omega}$

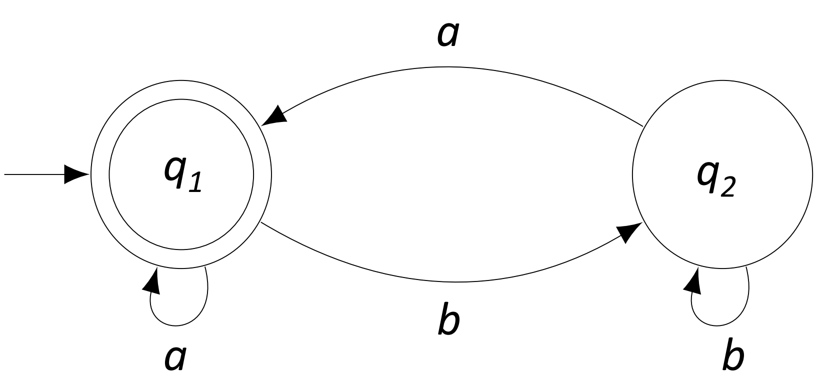
Automata on Infinite Words (Büchi)

$$\mathcal{B} = (\Sigma, Q, \Delta, Q^0, F)$$

- ρ is **accepting** $\Leftrightarrow$ $\mathbf{inf(\rho) \cap F \neq \emptyset}$

- Language of Büchi Automaton $\mathcal{B}$

ToDo Can you express it in LTL?



$\mathcal{L}(\mathcal{B})$ = {words with an Infinite number of a's}

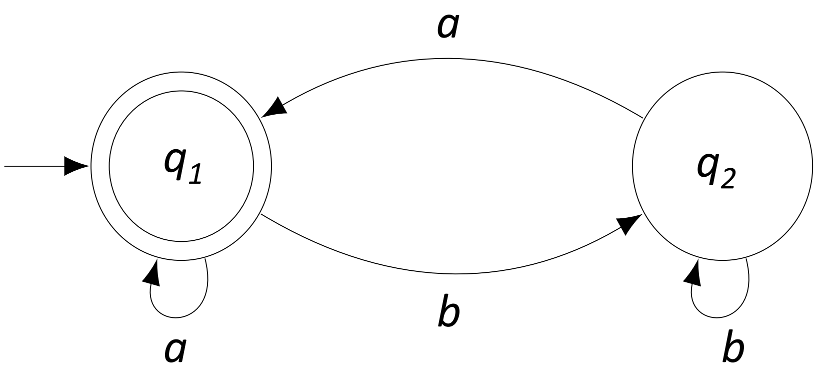
or

$\mathcal{L}(\mathcal{B}) = (\{a,b\}^*a)^\omega$

Automata on Infinite Words (Büchi)

$\mathcal{B} = (\Sigma, Q, \Delta, Q^0, F)$

- ρ is **accepting** $\Leftrightarrow$ $\mathbf{inf(\rho)} \cap \mathbf{F} \neq \emptyset$
- Language of Büchi Automaton $\mathcal{B}$



$\mathcal{L}(\mathcal{B})$ = {words with an Infinite number of a's}

or

$\mathcal{L}(\mathcal{B}) = (\{a,b\}^* a)^\omega$

In LTL: *GF(a)*

Outline

- Finite automata on finite words
- Automata on infinite words (Büchi automata)
- Deterministic vs non-deterministic Büchi automata
- Intersection of Büchi automata
- Checking emptiness of Büchi automata
- Generalized Büchi automata
- Model checking using automata

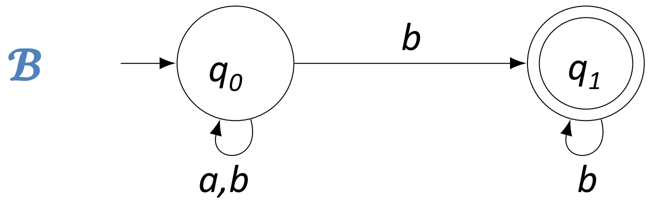
Det. and Non-det. Büchi Automata

- Deterministic Büchi automata are strictly less expressive than nondeterministic ones.
  - That is, not every nondeterministic Büchi automaton has an equivalent deterministic Büchi one.

Det. and Non-det. Büchi Automata

Theorem: There exists a non-deterministic Büchi automaton $\mathcal{B}$ for which there is no equivalent deterministic one.

Consider $\mathcal{B}$ below. What is its language? (Also in LTL)

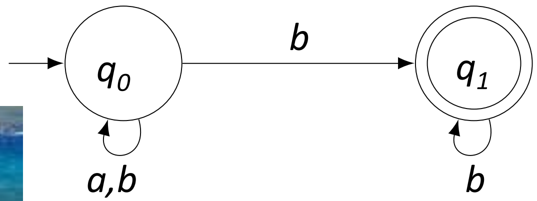


Det. and Non-det. Büchi Automata

Theorem: There exists a non-deterministic Büchi automaton $\mathcal{B}$ for which there is no equivalent deterministic one.

Consider $\mathcal{B}$ below. What is its language?

$\mathcal{B}$



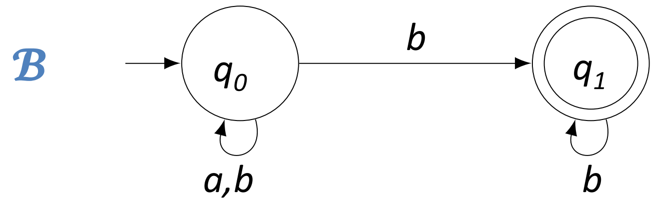
$\mathcal{L}(\mathcal{B})$ = {words with a **finite** number of a's}
or
$\mathcal{L}(\mathcal{B}) = \{a,b\}^* b^\omega$

In LTL :
FG¬a or **FG**b

Det. and Non-det. Büchi Automata

Theorem: There exists a non-deterministic Büchi automaton $\mathcal{B}$ for which there is no equivalent deterministic one.

Proof: The proof shows that there is no det. Büchi Automaton for "**finitely many**". Detailed proof see book.



$\mathcal{L}(\mathcal{B})$ = {words with a finite number of a's}
or
$\mathcal{L}(\mathcal{B}) = \{a,b\}^* b^\omega$

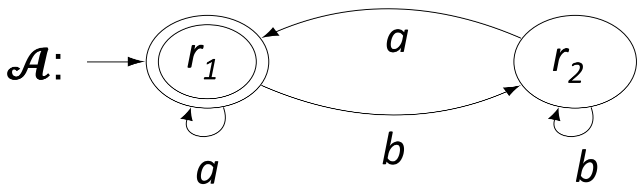
In LTL :
FG¬a or **FG**b

Det. and Non-det. Büchi Automata

Lemma 2: Deterministic Büchi automata are not closed under complementation.

Proof: ToDo

- Why? Hint: Automata below

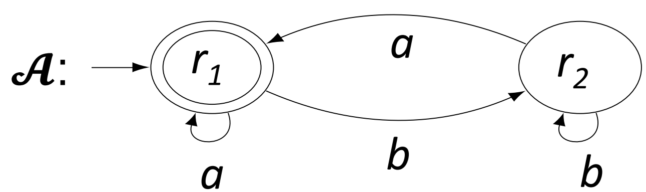


Det. and Non-det. Büchi Automata

Lemma 2: Deterministic Büchi automata are **not** closed under complementation.

Proof:

- Consider the language $\mathcal{L}$ = {words with infinitely many a's}.
- Construct a deterministic Büchi automaton $\mathcal{A}$ that accepts $\mathcal{L}$.
- Its complement is $\mathcal{L}$'={words with finitely many a's}, for which there is no deterministic Büchi automaton (see Theorem). □



Det. and Non-det. Büchi Automata

Theorem: Nondeterministic Büchi automata are closed under complementation.

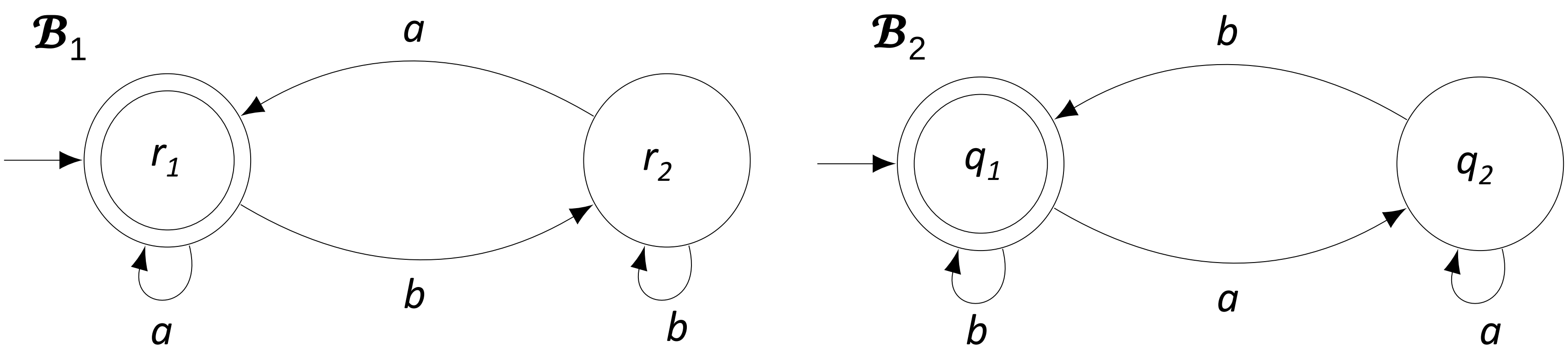
- The construction is very complicated. We will not see it here.
- Originally Büchi showed an algorithm for complementation that is double exponential in the size n of the automaton
- Mooly Safra (Tel-Aviv University) proved that it can be done by

$$2^{O(n \log n)}$$

Outline

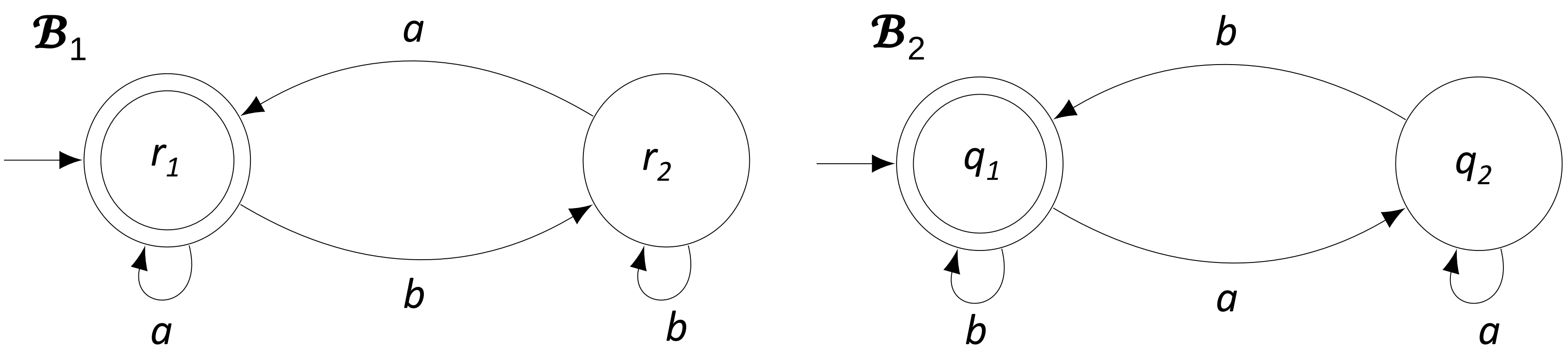
- Finite automata on finite words
- Automata on infinite words (Büchi automata)
- Deterministic vs non-deterministic Büchi automata
- **Intersection of Büchi automata**
- Checking emptiness of Büchi automata
- Generalized Büchi automata
- Model checking using automata

Intersection of Büchi Automata



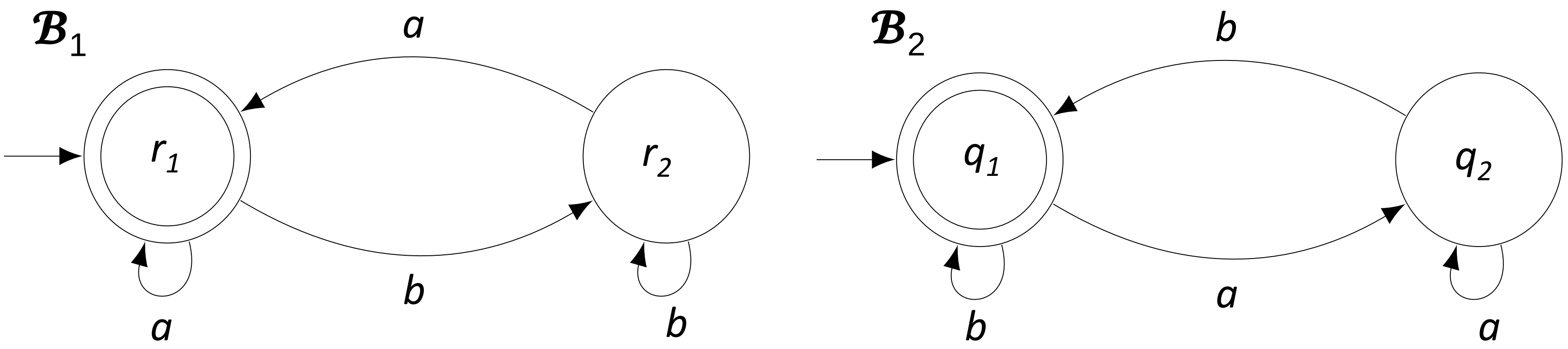
- What is $\mathcal{L}(\mathcal{B}_1) \cap \mathcal{L}(\mathcal{B}_2)$?

Intersection of Büchi Automata



- $\mathcal{L}(\mathcal{B}_1) \cap \mathcal{L}(\mathcal{B}_2)$ =
  {words with an infinite number of a’s and infinite number of b’s}
  (not empty)

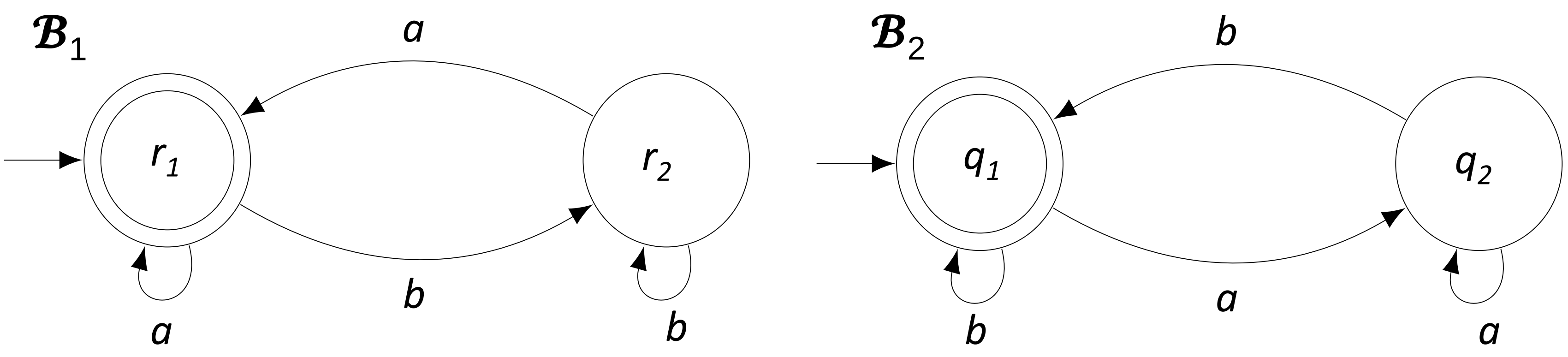
Intersection of Büchi Automata



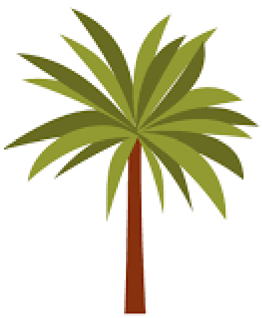
- $\mathcal{L}(\mathcal{B}_1) \cap \mathcal{L}(\mathcal{B}_2)$ =
  {words with an infinite number of a's and infinite number of b's}

- What do you get if you build the standard intersection?

Intersection of Büchi Automata



- $\mathcal{L}(\mathcal{B}_1) \cap \mathcal{L}(\mathcal{B}_2)$ =
  {words with an infinite number of a's and infinite number of b's}
- A standard intersection does not work – the automaton will not have any accepting states!
- **Solution: Introduce counter!**



Intersection of Büchi Automata

- Given $\mathcal{B}_1 = (\Sigma, Q_1, \Delta_1, Q_1^0, F_1)$ and $\mathcal{B}_2 = (\Sigma, Q_2, \Delta_2, Q_2^0, F_2)$
- $\mathcal{B} = (\Sigma, Q, \Delta, Q^0, F)$ s.t. $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{B}_1) \cap \mathcal{L}(\mathcal{B}_2)$ is defined as follows:
  - $Q = Q_1 \times Q_2 \times \{0, 1, 2\}$
  - $Q^0 = Q_1^0 \times Q_2^0 \times \{0\}$
  - $F = Q_1 \times Q_2 \times \{2\}$

Intersection of Büchi Automata

$((q_1,q_2,x), a, (q'_1,q'_2,x')) \in \boldsymbol{\Delta} \Leftrightarrow$

(1) $(q_1,a,q'_1) \in \boldsymbol{\Delta}_1$ and $(q_2,a,q'_2) \in \boldsymbol{\Delta}_2$ and

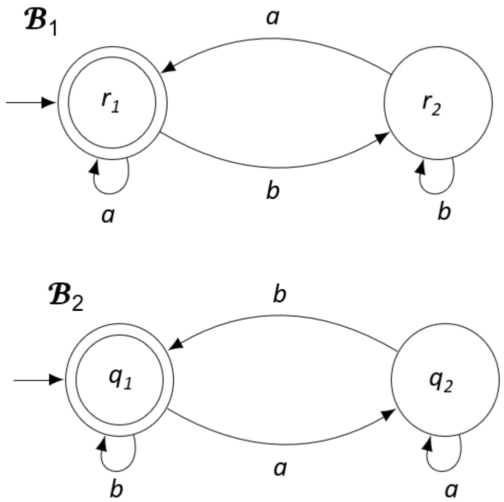
(2) If $x=0$ and $q'_1 \in \mathbf{F}_1$ then $x'=1$
If $x=1$ and $q'_2 \in \mathbf{F}_2$ then $x'=2$
If $x=2$ then $x'=0$
Else, $x'=x$

Explanation: $x=0$ is waiting for an accepting state from $\mathbf{F}_1$

$x=1$ is waiting for an accepting state from $\mathbf{F}_2$

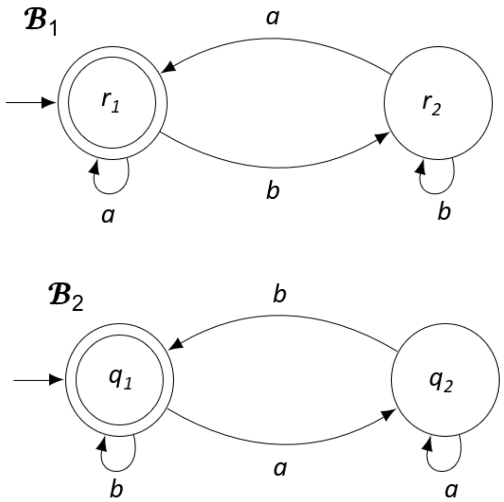
Intersection of Büchi Automata

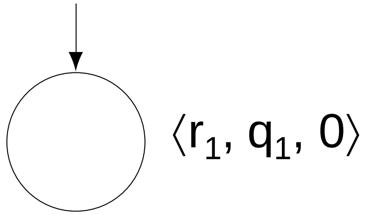
- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn



Intersection of Büchi Automata

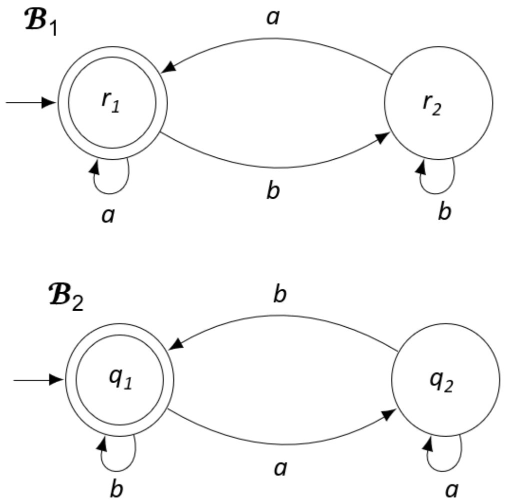
- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn

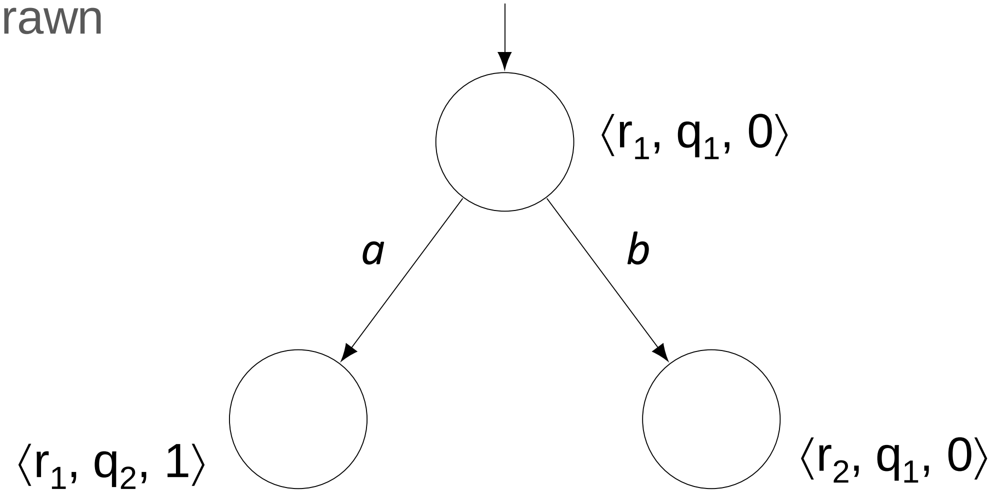




Intersection of Büchi Automata

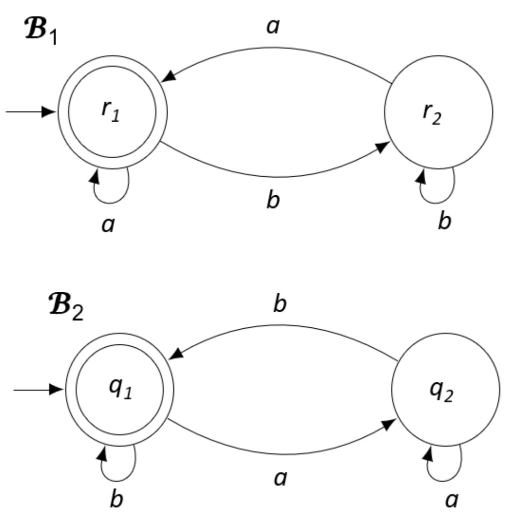
- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn

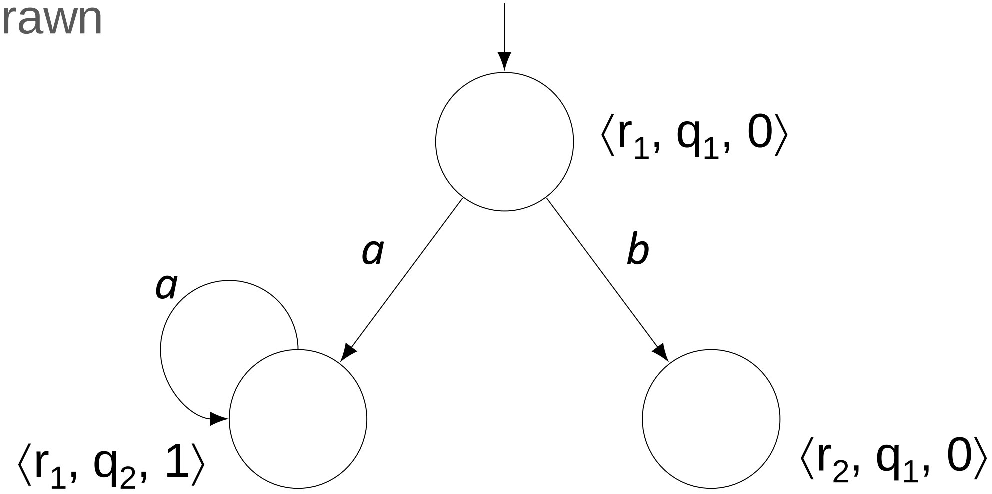




Intersection of Büchi Automata

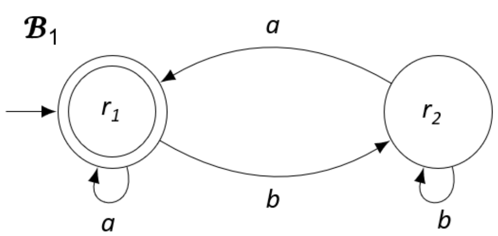
- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn

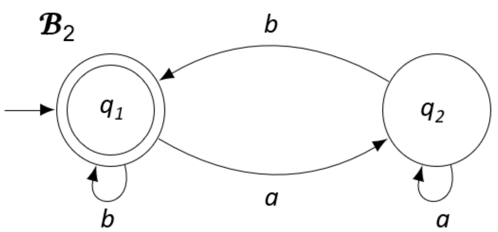


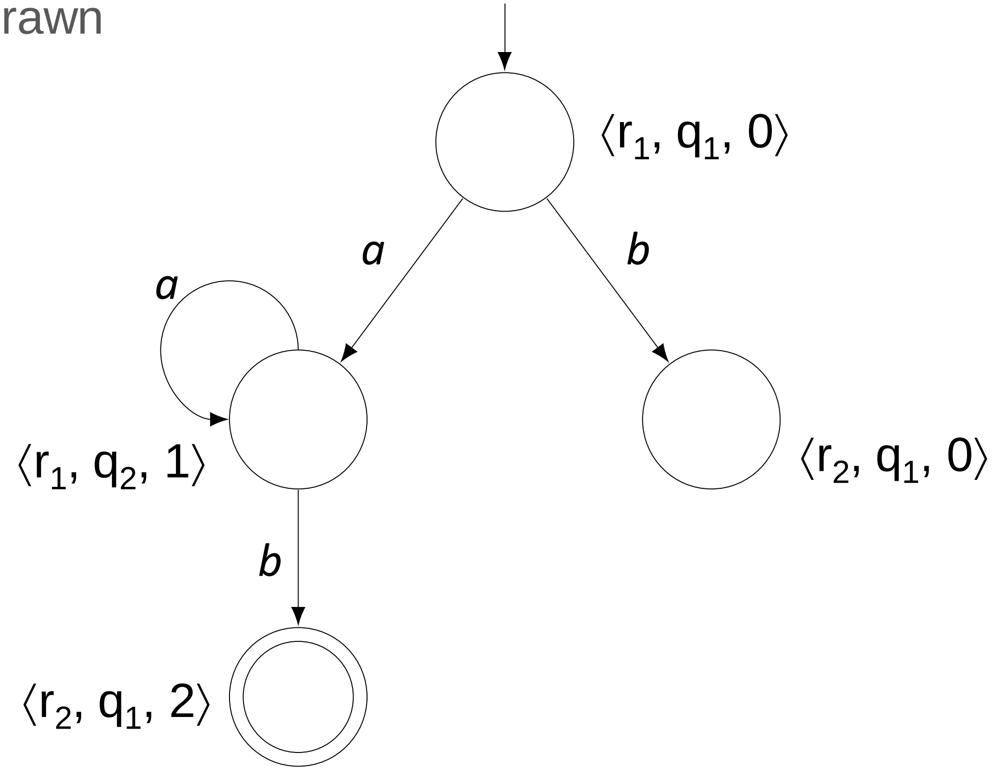


Intersection of Büchi Automata

- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn

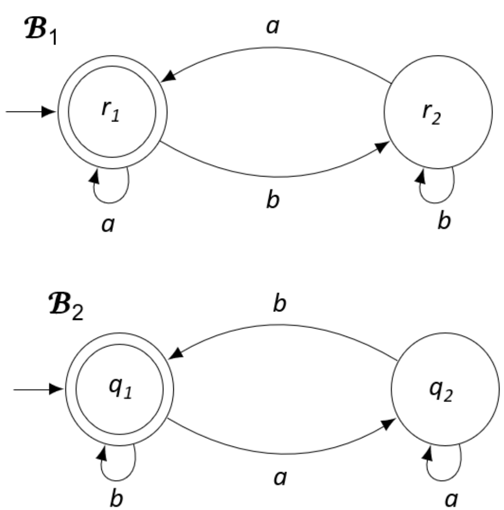


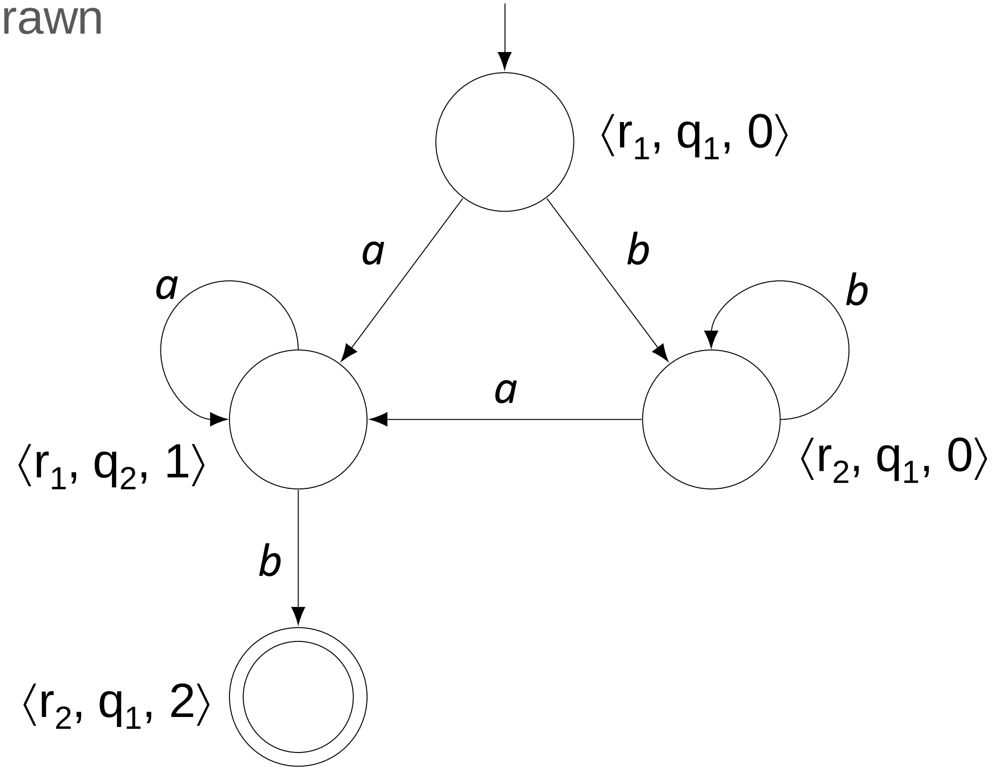




Intersection of Büchi Automata

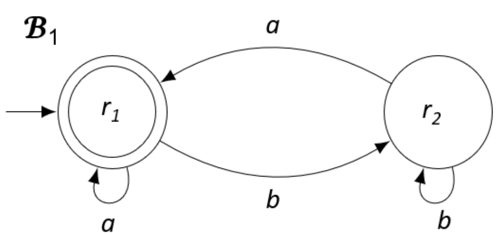
- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn

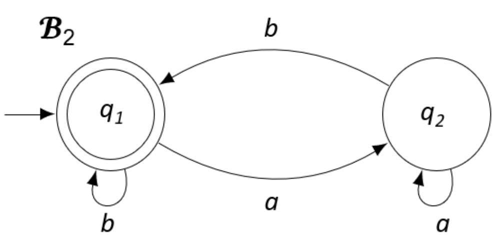


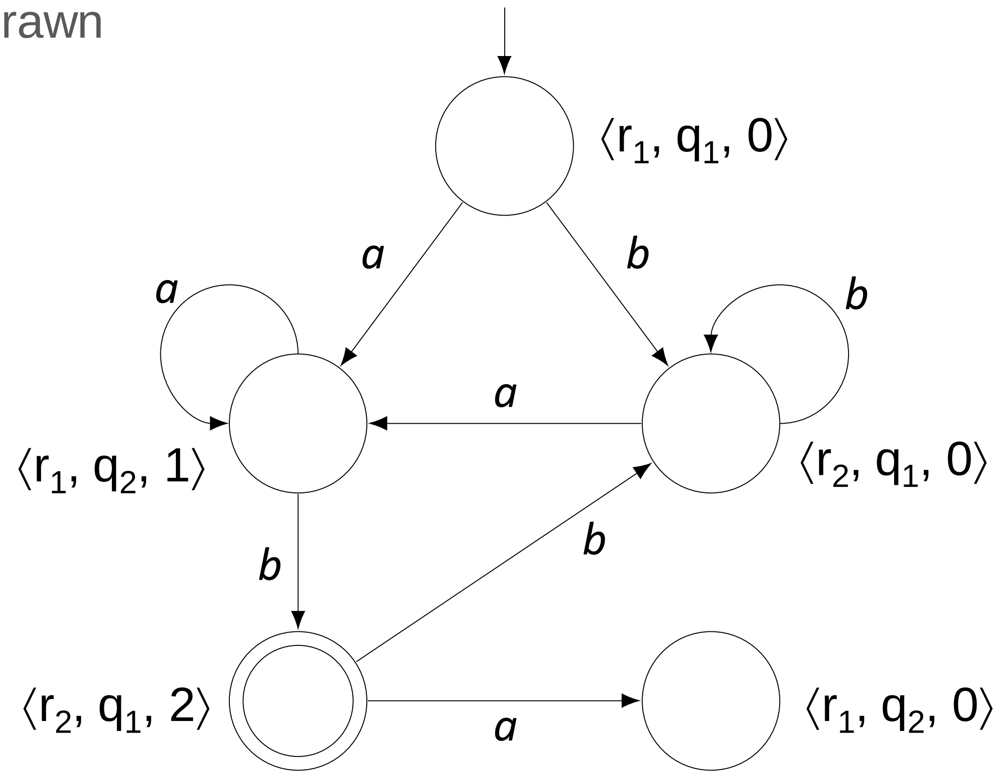


Intersection of Büchi Automata

- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn

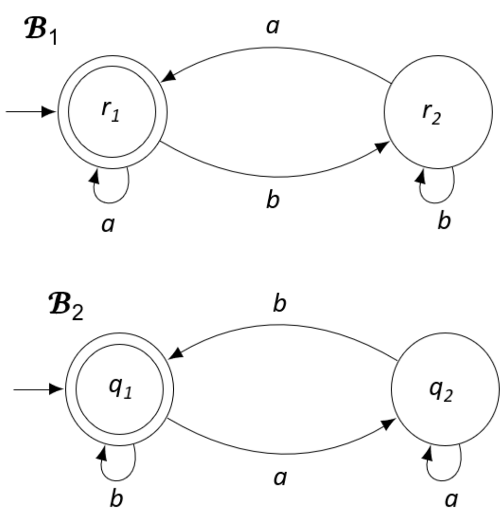


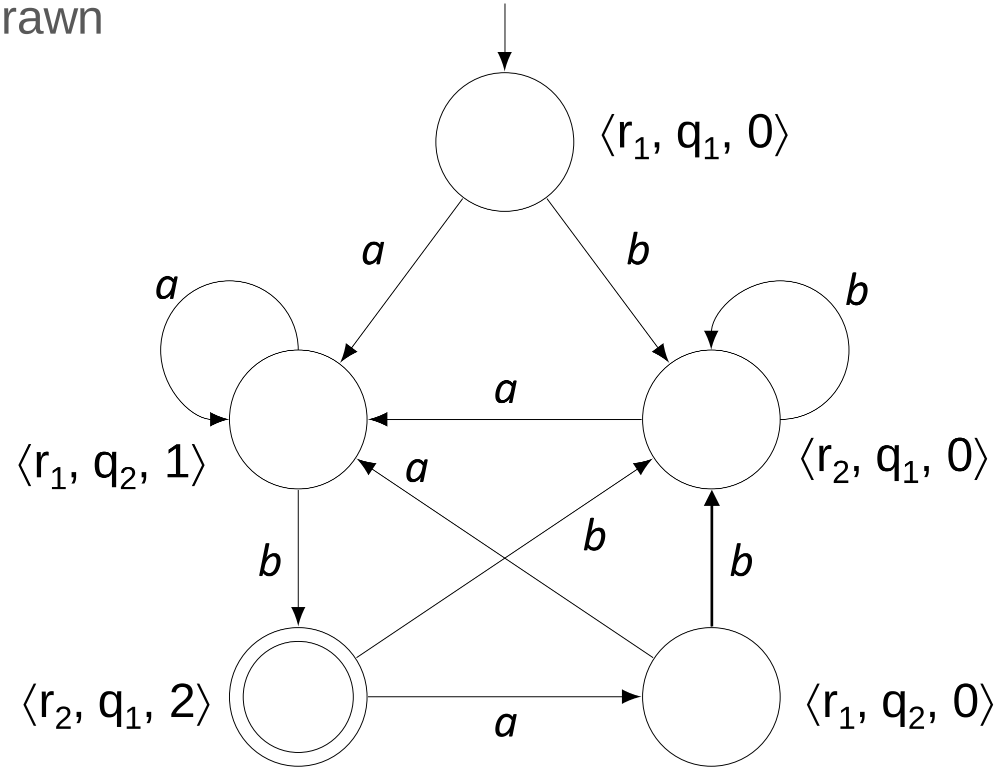




Intersection of Büchi Automata

- The first copy waits for an accepting state of $\mathcal{B}_1$
- The second copy waits for an accepting state of $\mathcal{B}_2$
- All states in the third copy are accepting
- Only the reachable states are drawn





Intersection of Büchi Automata

- Question
  - How do we define the transition relation for $\mathcal{B}$, if x is over {0,1} only?

With x over {0,1,2} we had:

$((q_1,q_2,x), a, (q'_1,q'_2,x')) \in \Delta \Leftrightarrow$

(1) $(q_1,a,q'_1) \in \Delta_1$ and $(q_2,a,q'_2) \in \Delta_2$ and

(2) If $x=0$ and $q'_1 \in F_1$ then $x'=1$
If $x=1$ and $q'_2 \in F_2$ then $x'=2$
If $x=2$ then $x'=0$
Else, $x'=x$

SCOS Secure & Correct Systems

Intersection of Büchi Automata

- Question
  - How do we define the transition relation for $\mathcal{B}$, if x is over {0,1} only?
- Answer
  - For $\mathbf{\Delta}$
    - (2) If x=0 and $q_1 \in \mathbf{F}_1$ then x'=1
      If x=1 and $q_2 \in \mathbf{F}_2$ then x'=0
      Else, x'=x
  - For $\mathbf{F}$
    - $\mathbf{F} = \mathbf{F}_1 \times \mathbf{Q}_2 \times \{0\}$

Intersection of Büchi Automata

- Question
  - In every interval we first wait for $\mathbf{F}_1$ and then wait for $\mathbf{F}_2$.
  - We ignore accepting states that don’t appear in this order.
  - Might we miss accepting paths in $\mathcal{B}$?

Intersection of Büchi Automata

- Question
  - In every interval we first wait for $\mathbf{F}_1$ and then wait for $\mathbf{F}_2$.
  - We ignore accepting states that don't appear in this order.
  - Might we miss accepting paths in $\mathcal{B}$?

- Answer
  - No. Since on an accepting path there are infinitely many of those, ignoring finite number of them in each interval will still lead us to the conclusion that the run is accepting

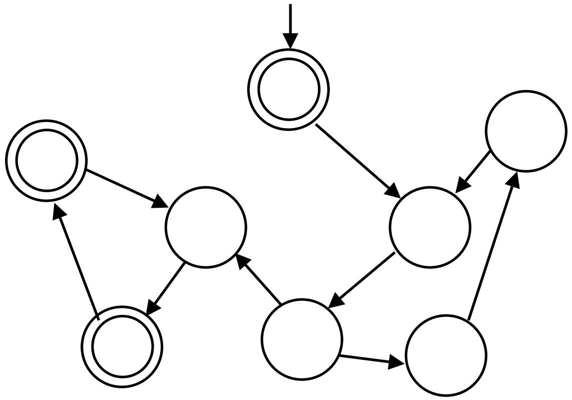
Outline

- Finite automata on finite words
- Automata on infinite words (Büchi automata)
- Deterministic vs non-deterministic Büchi automata
- Intersection of Büchi automata
- Checking emptiness of Büchi automata
- Generalized Büchi automata
- Automata and Kripke Structures
- Model checking using automata
- Translation of LTL to Büchi automata

Checking for emptiness of $\mathcal{L}(\mathcal{B})$

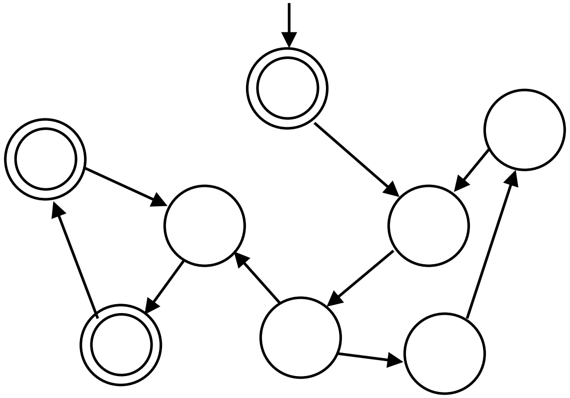
- An infinite run $\boldsymbol{\rho}$ is accepting ⇔ it visits an accepting state an infinite number of times.
  - **inf(ρ) ∩ $\mathbf{F} \neq \emptyset$**

How to check for **L(A) = ∅?**



Checking for emptiness of $\mathcal{L}(\mathcal{B})$

- An infinite run $\boldsymbol{\rho}$ is accepting $\Leftrightarrow$ it visits an accepting state an infinite number of times.
  - **$\text{inf}(\rho) \cap \mathbf{F} \neq \emptyset$**
- How to check for **L(A) = ∅?**
- Empty if there is no reachable accepting state on a cycle.

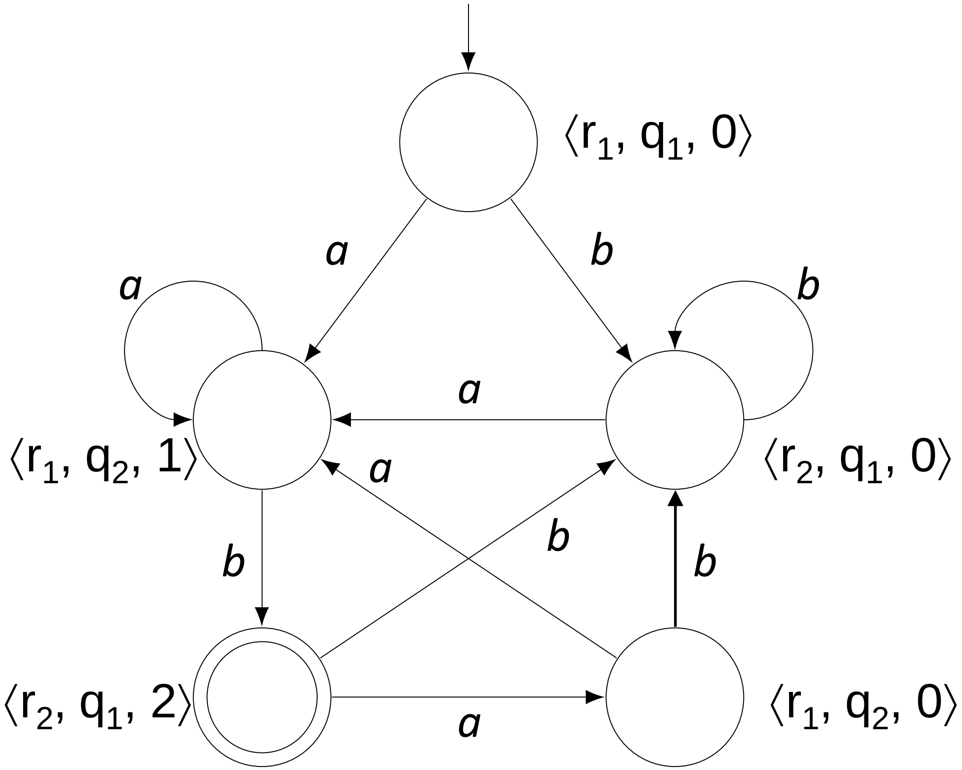


Non-emptiness ⇔ Existence of reachable accepting cycles

- *$\mathcal{L}(\mathcal{B})$ is nonempty ⇔*
- *The graph induced by $\mathcal{B}$ contains a path from an initial state of $\mathcal{B}$ to a state $\boldsymbol{t} \in \mathbf{F}$ and a path from $\boldsymbol{t}$ back to itself.*

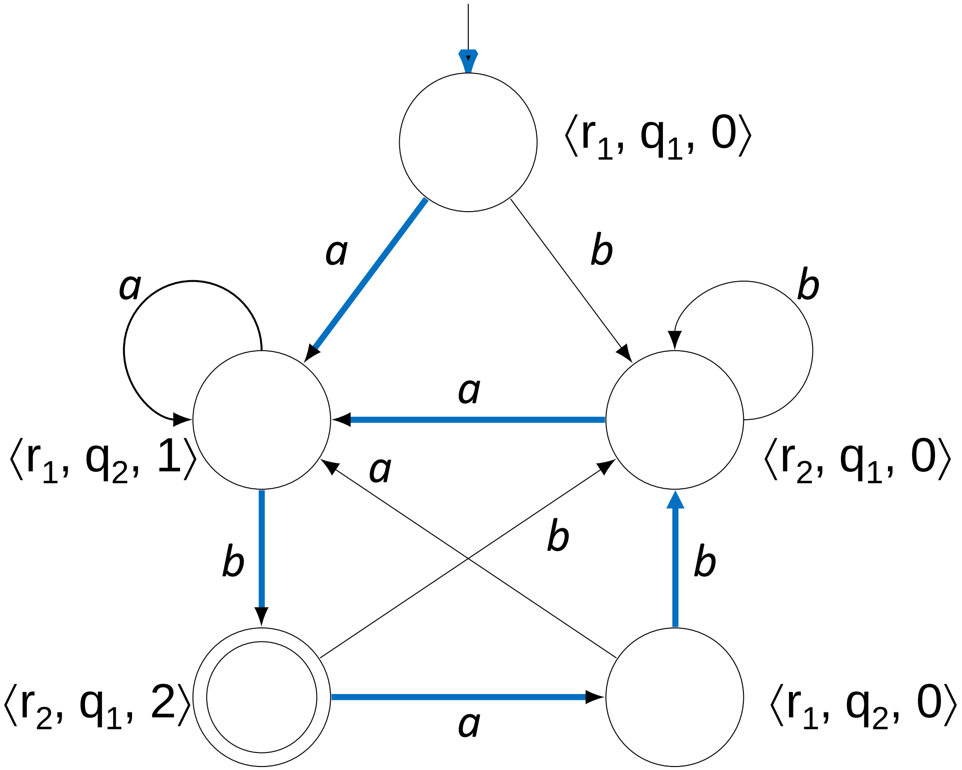
Example

- Is the language $\mathcal{L}(\mathcal{B})$ empty?



Example

- <r_2,q_1,2> is accepting and reachable from <r_1,q_1,0> and reachable from itself




THANK YOU!