Finite automata and formal languages (DIT323, TMV029)

Nils Anders Danielsson,
partly based on slides by Ana Bove

2024-01-25

Today

- Nondeterministic finite automata (NFAs).
- Equivalence of NFAs and DFAs.
- Perhaps something about how one can model things using finite automata.

The first assignment

In the first assignment you are given an inductively defined subset of $\{\, a, b \,\}^*$:

$$\frac{}{\varepsilon \in S} \qquad \frac{u, v \in S}{auavb \in S} \qquad \frac{u, v, w \in S}{buavaw \in S}$$

For this set we get the following induction principle (assuming "proof irrelevance"):

$$\begin{aligned}
&P(\varepsilon) \wedge \\
&(\forall u, v \in S. P(u) \wedge P(v) \Rightarrow P(auavb)) \wedge \\
&(\forall u, v, w \in S. P(u) \wedge P(v) \wedge P(w) \Rightarrow P(buavaw)) \\
&\quad \Rightarrow \\
&\forall w \in S. P(w)
\end{aligned}$$

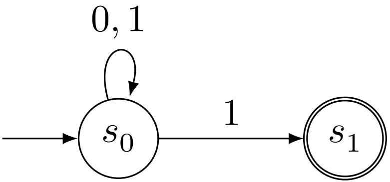
NFAs

NFAs

- Like DFAs, but multiple transitions may be possible.
- An NFA can be in multiple states at once.
- Can be easier to “program”.
- Can be much more compact.

NFAs

Strings over $\{0, 1\}$ that end with a one:



When a one is read the NFA "guesses" whether it should stay in s_0 or go to s_1.

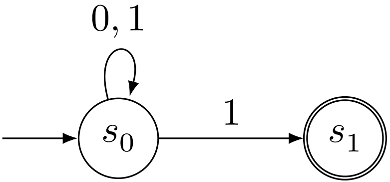
NFAs

An NFA can be given by a 5-tuple $(Q, \Sigma, \delta, q_0, F)$:

- A finite set of states (Q).
- An alphabet (Σ).
- A transition function ($\delta \in Q \times \Sigma \to \wp(Q)$).
- A start state ($q_0 \in Q$).
- A set of accepting states ($F \subseteq Q$).

NFAs

If the alphabet is $\{ 0, 1 \}$, then the diagram



corresponds to the 5-tuple

$$\mathit{Ends\text{-}with\text{-}one} = (\{ s_0, s_1 \}, \{ 0, 1 \}, \delta, s_0, \{ s_1 \}),$$

where δ is defined in the following way:

$$\begin{array}{ll} \delta \in \{ s_0, s_1 \} \times \{ 0, 1 \} \to \wp(\{ s_0, s_1 \}) & \\ \delta(s_0, 0) = \{ s_0 \} & \delta(s_1, \_) = \emptyset \\ \delta(s_0, 1) = \{ s_0, s_1 \} & \end{array}$$

The language of an NFA

The language $L(A)$ of an NFA $A = (Q, \Sigma, \gamma, q_0, F)$ is defined in the following way:

- A transition function for strings is defined by recursion:

$$\begin{aligned}
&\hat{\gamma} \in Q \times \Sigma^* \to \wp(Q) \\
&\hat{\gamma}(q, \varepsilon) \quad = \{ q \} \\
&\hat{\gamma}(q, aw) = \bigcup_{r \in \gamma(q,a)} \hat{\gamma}(r, w)
\end{aligned}$$

- The language is

$$\{ w \in \Sigma^* \mid \hat{\gamma}(q_0, w) \cap F \neq \emptyset \} .$$

The language of an NFA

For $Ends\text{-}with\text{-}one$:

$$\hat{\delta}(s_0, 10)$$

The language of an NFA

For *Ends-with-one*:

$$\hat{\delta}(s_0, 10) \quad =$$

$$\bigcup_{q \in \delta(s_0,1)} \hat{\delta}(q, 0)$$

The language of an NFA

For $\mathit{Ends\text{-}with\text{-}one}$:

$$\hat{\delta}(s_0, 10) \qquad = $$

$$\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0)$$

The language of an NFA

For *Ends-with-one*:

$$\hat{\delta}(s_0, 10) =$$

$$\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0) =$$

$$\bigcup_{q \in \{ s_0, s_1 \}} \bigcup_{r \in \delta(q,0)} \hat{\delta}(r, \varepsilon)$$

The language of an NFA

For *Ends-with-one*:

$$\begin{aligned}
&\hat{\delta}(s_0, 10) &=\\
&\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0) &=\\
&\bigcup_{q \in \{ s_0, s_1 \}} \bigcup_{r \in \delta(q,0)} \hat{\delta}(r, \varepsilon) &=\\
&\bigcup_{r \in \bigcup_{q \in \{ s_0, s_1 \}} \delta(q,0)} \hat{\delta}(r, \varepsilon)
\end{aligned}$$

The language of an NFA

For *Ends-with-one*:

$$
\begin{aligned}
\hat{\delta}(s_0, 10) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \bigcup_{r \in \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \bigcup_{q \in \{ s_0, s_1 \}} \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \delta(s_0,0) \cup \delta(s_1,0)} \hat{\delta}(r, \varepsilon)
\end{aligned}
$$

The language of an NFA

For *Ends-with-one*:

$$
\begin{aligned}
&\hat{\delta}(s_0, 10) &=\\
&\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0) &=\\
&\bigcup_{q \in \{ s_0, s_1 \}} \bigcup_{r \in \delta(q,0)} \hat{\delta}(r, \varepsilon) &=\\
&\bigcup_{r \in \bigcup_{q \in \{ s_0, s_1 \}} \delta(q,0)} \hat{\delta}(r, \varepsilon) &=\\
&\bigcup_{r \in \{ s_0 \} \cup \emptyset} \hat{\delta}(r, \varepsilon)
\end{aligned}
$$

The language of an NFA

For *Ends-with-one*:

$$\begin{aligned}
\hat{\delta}(s_0, 10) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \bigcup_{r \in \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \bigcup_{q \in \{ s_0, s_1 \}} \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \{ s_0 \}} \hat{\delta}(r, \varepsilon)
\end{aligned}$$

The language of an NFA

For *Ends-with-one*:

$$
\begin{aligned}
\hat{\delta}(s_0, 10) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \bigcup_{r \in \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \bigcup_{q \in \{ s_0, s_1 \}} \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \{ s_0 \}} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \{ s_0 \}} \{ r \}
\end{aligned}
$$

The language of an NFA

For *Ends-with-one*:

$$
\begin{aligned}
\hat{\delta}(s_0, 10) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \hat{\delta}(q, 0) &= \\
\bigcup_{q \in \{ s_0, s_1 \}} \bigcup_{r \in \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \bigcup_{q \in \{ s_0, s_1 \}} \delta(q,0)} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \{ s_0 \}} \hat{\delta}(r, \varepsilon) &= \\
\bigcup_{r \in \{ s_0 \}} \{ r \} &= \\
\{ s_0 \}
\end{aligned}
$$

Transition diagrams

As for DFAs, but with one change:

- For every transition $\delta(s_1, a) = S$
  and every state $s_2 \in S$,
  an arrow marked with a from s_1 to s_2.

Note:

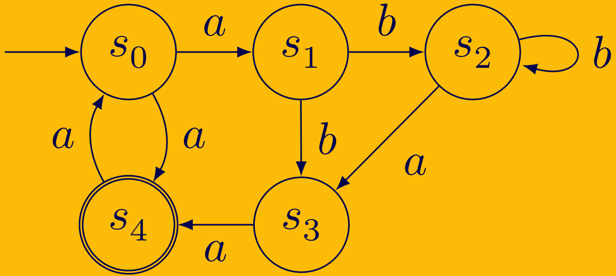
- The alphabet is not defined unambiguously.
- No need for special treatment of missing transitions, because $\delta(s_1, a)$ can be empty.

Transition tables

As for DFAs, but with one change:

- The result of a transition is a set of states instead of a state.

Which strings are members of the language of the following NFA over $\{ a, b, c \}$?



1. $abba$.
2. $abbaca$.
3. $aaabaa$.
4. $aaabaaa$.
5. $aaaabaa$.
6. $abbaaabaaa$.

Respond at `https://pingo.coactum.de/729558`.

Some conventions

At least partly following the course text book:

- q, r, s: A state.
- δ: A transition function.

Which of the following propositions are valid?

1. $\hat{\delta}(q, a) = \delta(q, a)$.
2. $\hat{\delta}(q, uv) = \hat{\delta}(q, vu)$.
3. $\hat{\delta}(q, uv) = \bigcup_{r \in \hat{\delta}(q,v)} \hat{\delta}(r, u)$.
4. $\hat{\delta}(q, uv) = \bigcup_{r \in \hat{\delta}(q,u)} \hat{\delta}(r, v)$.

You may want to use the following lemma:

$$\bigcup_{y \in \bigcup_{x \in X} F(x)} G(y) = \bigcup_{x \in X} \bigcup_{y \in F(x)} G(y)$$

Respond at `https://pingo.coactum.de/729558`.

Which of the following propositions are valid?

1. $\hat{\delta}(q, a) = \delta(q, a)$.

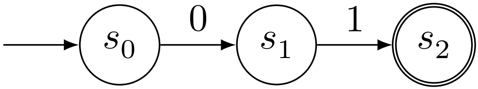
Yes:

$$\begin{aligned} \hat{\delta}(q, a) &= \\ \bigcup_{r \in \delta(q,a)} \hat{\delta}(r, \varepsilon) &= \\ \bigcup_{r \in \delta(q,a)} \{\, r \,\} &= \\ \delta(q, a) \end{aligned}$$

Which of the following propositions are valid?

2. $\hat{\delta}(q, uv) = \hat{\delta}(q, vu)$.

No. Counterexample:



Denote the transition function by δ.

$$\hat{\delta}(s_0, 01) = \{ s_2 \} \neq \emptyset = \hat{\delta}(s_0, 10)$$

Which of the following propositions are valid?

4. $\hat{\delta}(q, uv) = \bigcup_{r \in \hat{\delta}(q,u)} \hat{\delta}(r, v)$.

Yes. Proof by induction on the structure of the string u:

$$\begin{aligned}
&\hat{\delta}(q, \varepsilon v) &&= \\
&\hat{\delta}(q, v) &&= \\
&\bigcup_{r \in \{\, q \,\}} \hat{\delta}(r, v) &&= \\
&\bigcup_{r \in \hat{\delta}(q,\varepsilon)} \hat{\delta}(r, v)
\end{aligned}$$

Which of the following propositions are valid?

4. $\hat{\delta}(q, uv) = \bigcup_{r \in \hat{\delta}(q,u)} \hat{\delta}(r, v)$.

Yes. Proof by induction on the structure of the string u:

$$
\begin{aligned}
&\hat{\delta}(q, auv) &=\\
&\bigcup_{r' \in \delta(q,a)} \hat{\delta}(r', uv) &=\\
&\bigcup_{r' \in \delta(q,a)} \bigcup_{r \in \hat{\delta}(r',u)} \hat{\delta}(r, v) &=\\
&\bigcup_{r \in \bigcup_{r' \in \delta(q,a)} \hat{\delta}(r',u)} \hat{\delta}(r, v) &=\\
&\bigcup_{r \in \hat{\delta}(q,au)} \hat{\delta}(r, v)
\end{aligned}
$$

Which of the following propositions are valid?

3. $\hat{\delta}(q, uv) = \bigcup_{r \in \hat{\delta}(q,v)} \hat{\delta}(r, u)$.

No, we have

$$\bigcup_{r \in \hat{\delta}(q,v)} \hat{\delta}(r, u) = \hat{\delta}(q, vu),$$

which in general is not equal to $\hat{\delta}(q, uv)$.

NFAs versus DFAs

NFAs versus DFAs

- Every DFA can be seen as an NFA:
  - Turn $\delta(s_1, a) = s_2$ into $\delta(s_1, a) = \{ s_2 \}$.
- Thus every language that can be defined by a DFA can also be defined by an NFA.
- What about the other direction?
  Are NFAs more powerful?
- No.

Subset construction

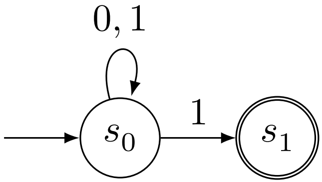
Given an NFA $N = (Q, \Sigma, \delta, q_0, F)$ we can define a DFA D with the same alphabet in such a way that $L(N) = L(D)$:

$$D = (\wp(Q), \Sigma, \delta', \{ q_0 \}, \{ S \subseteq Q \mid S \cap F \neq \emptyset \})$$
$$\delta'(S, a) = \bigcup_{s \in S} \delta(s, a)$$

- The DFA keeps track of exactly which states the NFA is in.
- It accepts if at least one of the NFA states is accepting.

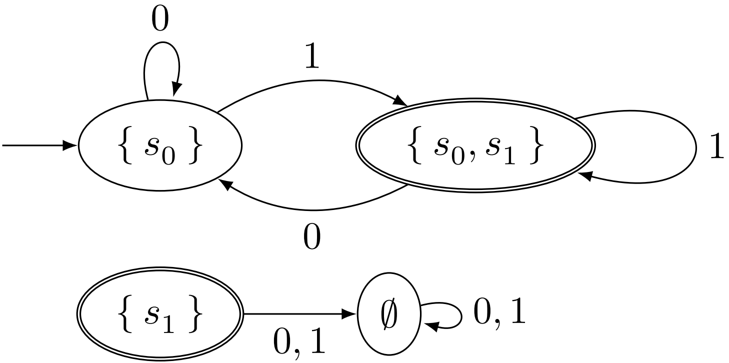
Subset construction

An NFA:



Subset construction

If we apply the subset construction we get the following DFA:

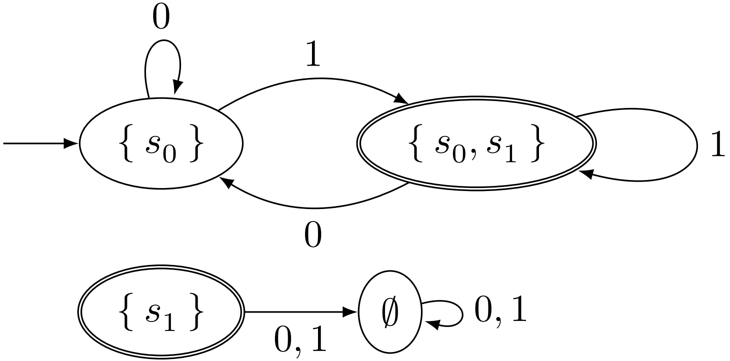


If an NFA has 10 states, and we use the subset construction to build a corresponding DFA, how many states does the DFA have?

Respond at `https://pingo.coactum.de/729558`.

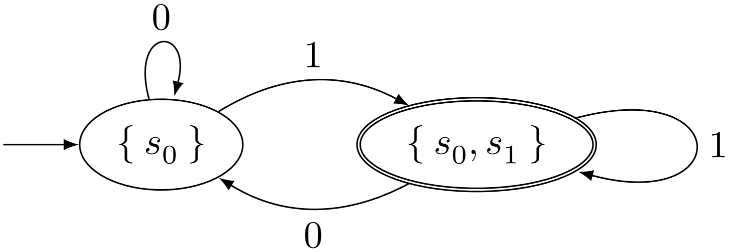
Accessible states

Note that some states cannot be reached from the start state:



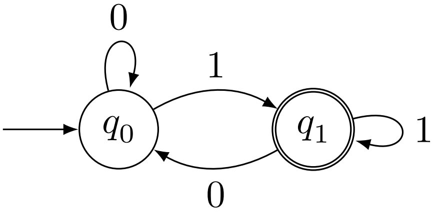
Accessible states

If we remove non-accessible states, then we get a DFA which defines the same language:



Accessible states

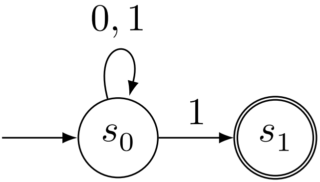
One can also rename the states:



Subset construction

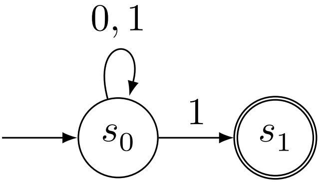
- Note that one does not have to first construct a DFA with $2^{|Q|}$ states, and then remove inaccessible states.
- One can instead construct the DFA without inaccessible states right away:
  - Start with the start state.
  - Add new states reachable from the start state.
  - Add new states reachable from those states.
  - And so on until there are no more new states.

Subset construction



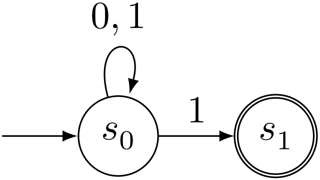
	0	1
$\rightarrow \{ s_0 \}$		

Subset construction



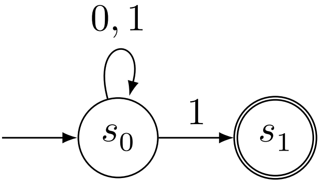
	0	1
$\rightarrow \{ s_0 \}$	$\{ s_0 \}$	

Subset construction



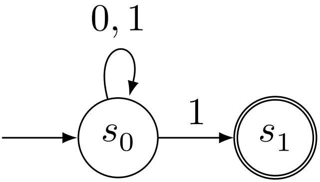
	0	1
$\rightarrow \{ s_0 \}$	$\{ s_0 \}$	$\{ s_0, s_1 \}$
$* \{ s_0, s_1 \}$		

Subset construction



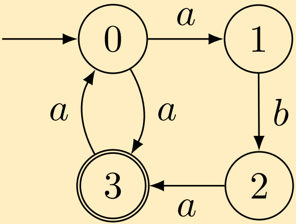
	0	1
$\rightarrow \{\, s_0 \,\}$	$\{\, s_0 \,\}$	$\{\, s_0, s_1 \,\}$
$* \{\, s_0, s_1 \,\}$	$\{\, s_0 \,\}$	

Subset construction



	0	1
$\rightarrow \{ s_0 \}$	$\{ s_0 \}$	$\{ s_0, s_1 \}$
$* \{ s_0, s_1 \}$	$\{ s_0 \}$	$\{ s_0, s_1 \}$

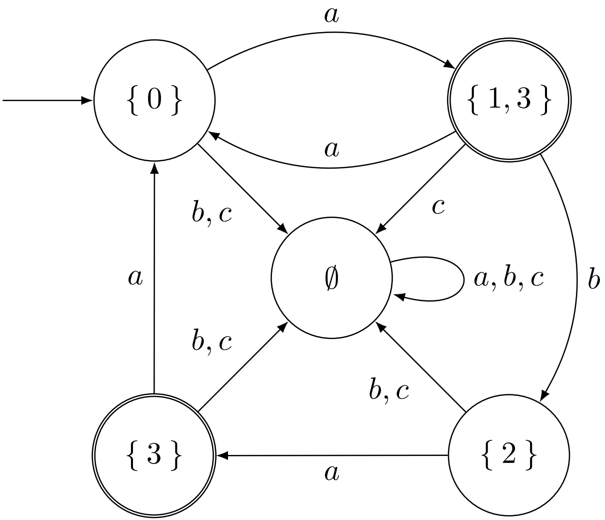
If the subset construction is used to build a DFA corresponding to the following NFA over $\{ a, b, c \}$, and inaccessible states are removed, how many states are there in the resulting DFA?



Respond at `https://pingo.coactum.de/729558`.

How many states are there in the resulting DFA?

5:



Subset construction

Recall the subset construction for $N = (Q, \Sigma, \delta, q_0, F)$:

$$D = (\wp(Q), \Sigma, \delta', \{ q_0 \}, \{ S \subseteq Q \mid S \cap F \neq \emptyset \})$$
$$\delta'(S, a) = \bigcup_{s \in S} \delta(s, a)$$

How would you prove $L(N) = L(D)$?

$$L(N) = \left\{ w \in \Sigma^* \;\middle|\; \hat{\delta}(q_0, w) \cap F \neq \emptyset \right\}$$
$$L(D) = \left\{ w \in \Sigma^* \;\middle|\; \begin{array}{l} \widehat{\delta'}(\{ q_0 \}, w) \in \\ \qquad \{ S \subseteq Q \mid S \cap F \neq \emptyset \} \end{array} \right\}$$

Subset construction

Recall the subset construction for $N = (Q, \Sigma, \delta, q_0, F)$:

$$D = (\wp(Q), \Sigma, \delta', \{ q_0 \}, \{ S \subseteq Q \mid S \cap F \neq \emptyset \})$$
$$\delta'(S, a) = \bigcup_{s \in S} \delta(s, a)$$

How would you prove $L(N) = L(D)$?

$$L(N) = \left\{ w \in \Sigma^* \;\middle|\; \hat{\delta}(q_0, w) \cap F \neq \emptyset \right\}$$
$$L(D) = \left\{ w \in \Sigma^* \;\middle|\; \widehat{\delta'}(\{ q_0 \}, w) \cap F \neq \emptyset \right\}$$

Subset construction

This follows from

$$\forall w \in \Sigma^*.\ \forall q \in Q.\ \hat{\delta}(q, w) = \widehat{\delta'}(\{\ q\ \}, w),$$

which can be proved by induction on the structure of the string, using the following lemma:

$$\forall w \in \Sigma^*.\ \forall S \subseteq Q.\ \widehat{\delta'}(S, w) = \bigcup_{s \in S} \widehat{\delta'}(\{\ s\ \}, w)$$

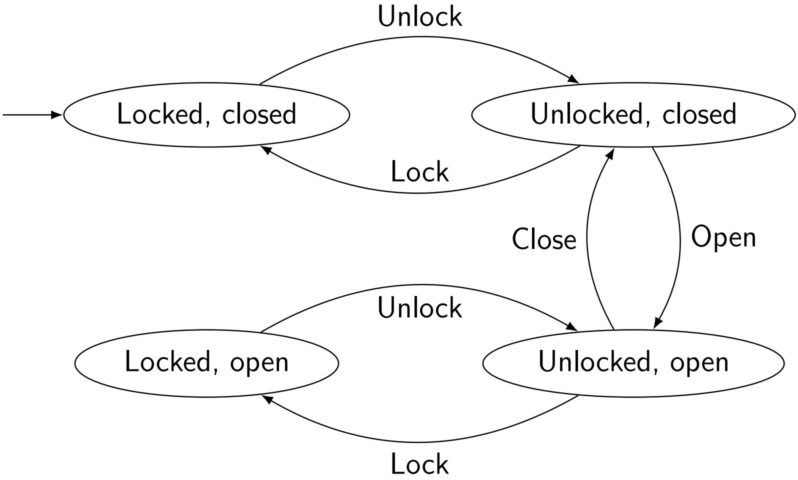
The lemma can also be proved by induction on the structure of the string.

Regular languages

- Recall that a language $M \subseteq \Sigma^*$ is regular if there is some DFA A with alphabet Σ such that $L(A) = M$.
- A language $M \subseteq \Sigma^*$ is also regular if there is some *NFA* A with alphabet Σ such that $L(A) = M$.

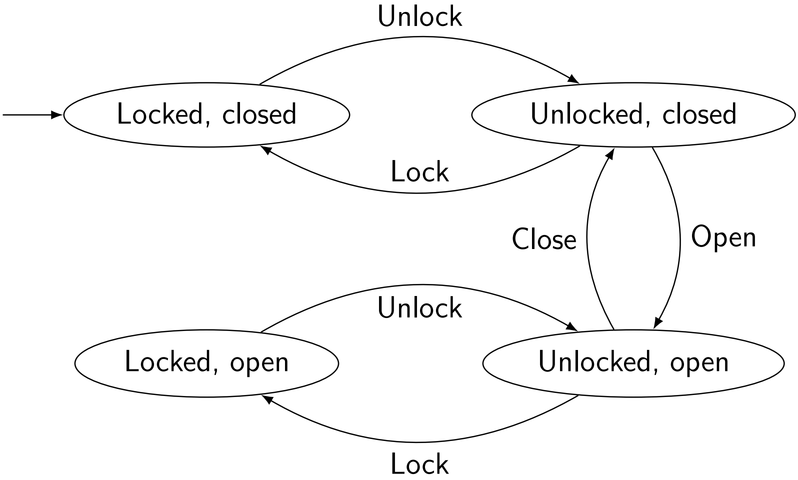
Models

A model of a door



Alphabet: { Lock, Unlock, Open, Close }.

A model of a door



What happens if we try to lock a locked door? Does the system “crash”?

Try to model something as a finite automaton:

- The traffic lights of an intersection.
- A coin-operated vending machine.
- ...

How well does your model work? Does it make sense to model the phenomenon as a finite automaton?

Today

- Nondeterministic finite automata (NFAs).
- The subset construction.
- Models.

Consultation time

- Tomorrow.
- You decide what you want to work on.

Next lecture

Nondeterministic finite automata with ε-transitions.