Finite automata theory and formal languages (DIT321, TMV027)

Nils Anders Danielsson,
partly based on slides by Ana Bove

2019-01-31

Today

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

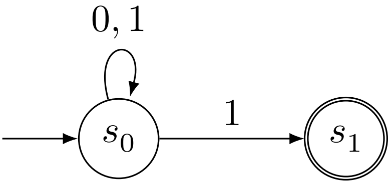
NFAs

NFAs

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

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 ($F \subseteq Q$).

The language of an NFA

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

- A transition function for strings is defined by recursion:

$$\hat{\delta} \in Q \times \Sigma^* \to \wp(Q)$$

$$\hat{\delta}(q, \varepsilon) = \{ q \}$$

$$\hat{\delta}(q, aw) = \bigcup_{r \in \delta(q,a)} \hat{\delta}(r, w)$$

- The language is

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

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

Transition diagrams

As for DFAs, but with one change:

- For every transition $\delta(q, a) = S$, an arrow marked with a from q to every node in S.

Note:

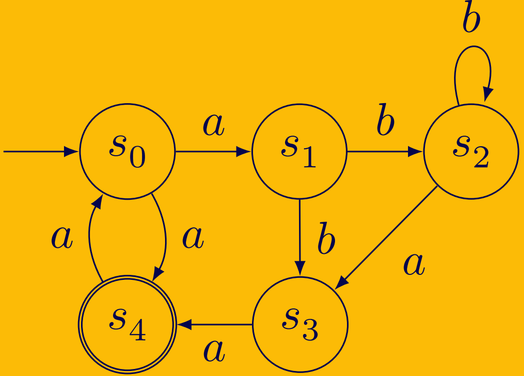
- The alphabet is not defined unambiguously.
- No need for special treatment of missing transitions, because $\delta(q, a)$ can be empty.

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. $abbaaabaaa$.

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

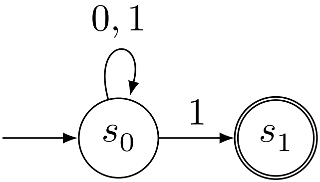
Given an NFA $N = (Q, \Sigma, \delta, q_0, F)$ we can define a DFA D with the same alphabet in such a way that $L(N) = L(D)$:

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

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

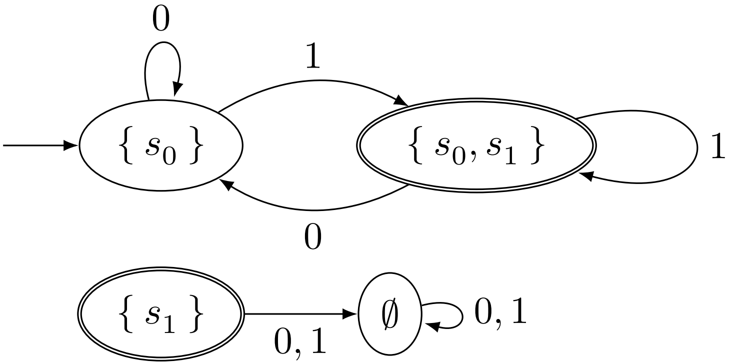
Subset construction

An NFA:



Subset construction

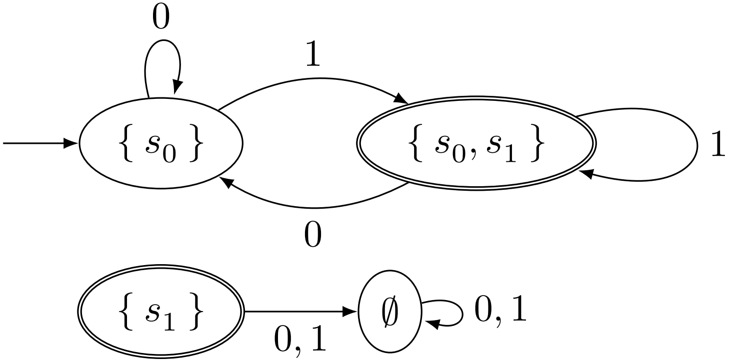
If we apply the subset construction we get the following DFA:



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

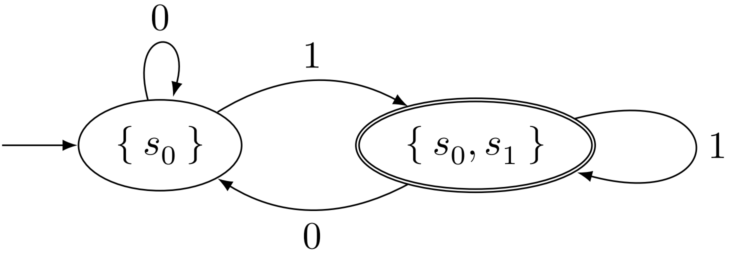
Accessible states

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



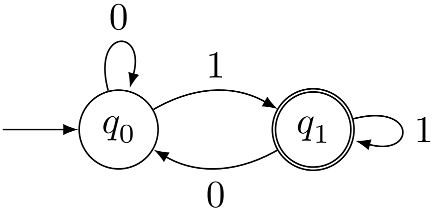
Accessible states

The following DFA defines the same language:



Accessible states

One can also rename the states:



Accessible states

- Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA.
- The set $Acc(q) \subseteq Q$ of states that are accessible from $q \in Q$ can be defined in the following way:

$$Acc(q) = \left\{ \hat{\delta}(q, w) \;\middle|\; w \in \Sigma^* \right\}$$

- A possibly smaller DFA:

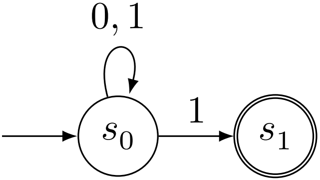
$$A' = (Acc(q_0), \Sigma, \delta', q_0, F \cap Acc(q_0))$$
$$\delta'(q, a) = \delta(q, a)$$

- We have $L(A') = L(A)$.

Subset construction

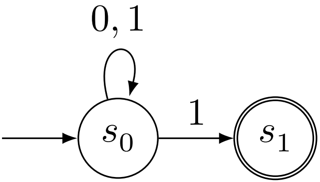
- Note that one does not have to first construct a DFA with $2^{|Q|}$ states, and then remove inaccessible states.
- One can instead construct the DFA without inaccessible states right away:
  - Start with the start state.
  - Add new states reachable from the start state.
  - Add new states reachable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

Subset construction



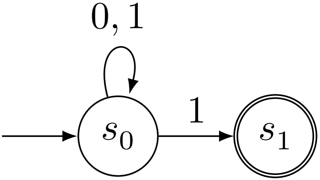
	0	1
$\rightarrow \{ s_0 \}$		

Subset construction



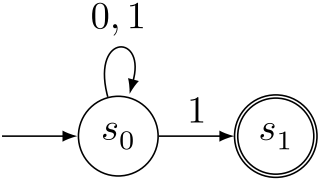
	0	1
$\rightarrow \{ s_0 \}$	$\{ s_0 \}$	

Subset construction



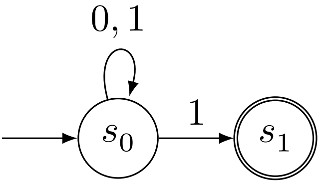
	0	1
$\to \{ s_0 \}$	$\{ s_0 \}$	$\{ s_0, s_1 \}$
$* \{ s_0, s_1 \}$		

Subset construction



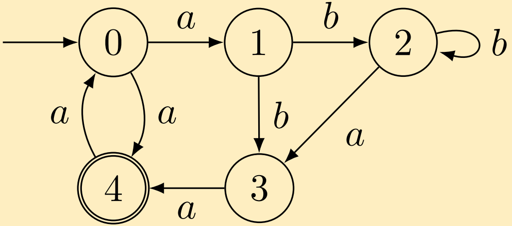
	0	1
$\rightarrow \{ s_0 \}$	$\{ s_0 \}$	$\{ s_0, s_1 \}$
$* \{ s_0, s_1 \}$	$\{ s_0 \}$	

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?



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:

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

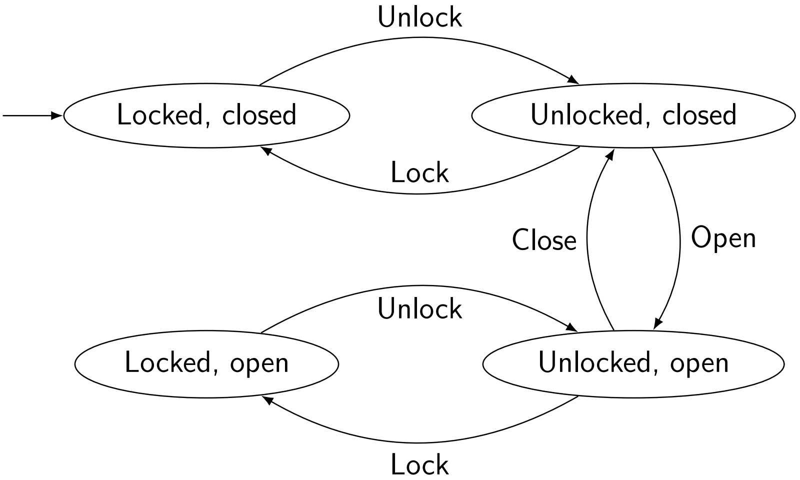
The lemma can also be proved by induction on the structure of a string.

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 $L(A) = M$.

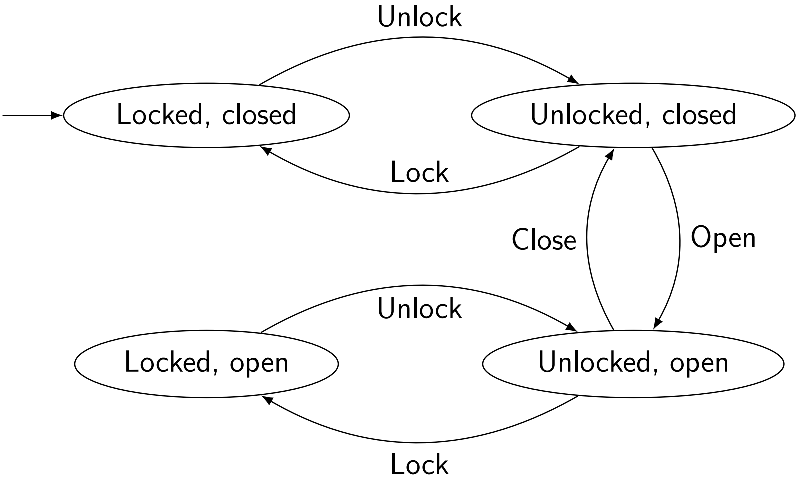
Models

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.
- Accessible states.
- Models.

Next lecture

Nondeterministic finite automata with ε-transitions.

- Deadline for the next quiz: 2019-02-04, 10:00.
- Deadline for the first assignment: 2019-02-03, 23:59.