Automata Theoretic LTL Model Checking

Giuseppe Perelli

SAPIENZA
UNIVERSITÀ DI ROMA

Formal Methods 2020/21

- An alphabet is a (finite) set of symbols (letters). E.g. $\Sigma = \{a, b\}$
- A finite word over Σ is a finite sequence of letters. E.g. $w = ababbab$
- An infinite word over Σ is an infinite sequence of letters. E.g. $w = ababbab\ldots$
- The sets of all finite and infinite words are denoted Σ^* and Σ^ω, respectively.
- A finite language is a subset $L \subseteq \Sigma^*$. E.g. $L =$ "words ending with an a"
- An infinite language is a subset $L \subseteq \Sigma^\omega$. E.g. $L =$ "words containing a finite number of a"

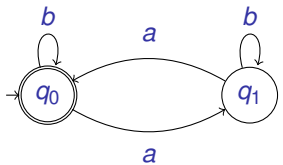
Language recognition problem

Determine whether a word w belongs to a language L.

Automata used as computational devices used to solve language recognition and related problems.

A Deterministic Finite-State Automaton (DFA) is a tuple $\mathcal{D} = \langle Q, \Sigma, s, \delta, F \rangle$ with:

- Q finite set of states
- Σ finite alphabet
- $s \in Q$ initial state
- $F \subseteq Q$ set of final states
- $\delta : Q \times \Sigma \to Q$ transition function



The automaton recognizes the words with an even number of a.

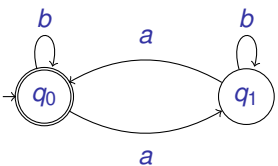
A word $w \in \Sigma^*$ is read on $\mathcal{D}$ by starting from s and following the transition function, generating a run $\rho \in Q^*$.
We say $w \in \mathcal{L}(\mathcal{D}) \subseteq \Sigma^*$ if the corresponding run ρ ends in a final state.

Sample execution:

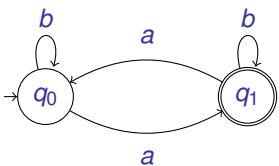
$$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_1 \xrightarrow{a} q_0 \xrightarrow{b} q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_1$$

For a given $\mathcal{D} = \langle Q, \Sigma, q_0, \delta, F \rangle$, the dual automaton $\overline{\mathcal{D}} = \langle Q, \Sigma, q_0, \delta, Q \setminus F \rangle$ is such that

$$\mathcal{L}(\overline{\mathcal{D}}) = \Sigma^* \setminus \mathcal{L}(\mathcal{D})$$



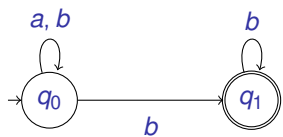
Recognizes words with an even number of a.



Recognizes words with an odd number of a.

A Nondeterministic Finite-State Automaton (NFA) is a tuple $\mathcal{N} = \langle Q, \Sigma, I, \delta, F \rangle$ with:

- Q finite set of states
- Σ finite alphabet
- $I \subseteq Q$ set of initial states
- $F \subseteq Q$ set of final states
- $\delta : Q \times \Sigma \to 2^Q$ nondeterministic transition function



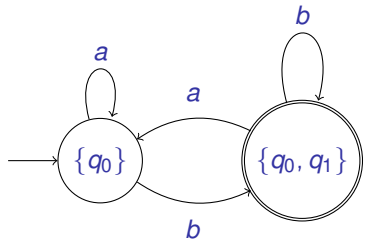
The automaton recognizes the words that end with b.

More than one run is possible on the same word $w \in \Sigma^*$.
The automaton $\mathcal{N}$ accepts $w \in \mathcal{L}(\mathcal{N})$ if at least one run is accepting.

The subset construction

Let $\mathcal{N} = \langle Q, \Sigma, I, \delta, F \rangle$ be a nondeterministic automaton. Consider the deterministic automaton $\mathcal{D}_{\mathcal{N}} = \langle 2^Q, \Sigma, Q_0, \delta', \mathcal{F} \rangle$ with:

- $Q_0 = I$
- $\mathcal{F} = \{Q' \subseteq Q : Q' \cap F \neq \emptyset\}$
- $\delta'(Q', \sigma) = \bigcup_{q \in Q'} \delta(q, \sigma)$



Intuitively, the automaton $\mathcal{D}_{\mathcal{N}}$ runs all the possible executions of $\mathcal{N}$ in parallel.
If one of them accepts the word in $\mathcal{N}$, then it does so in $\mathcal{D}_{\mathcal{N}}$. $\mathcal{L}(\mathcal{D}_{\mathcal{N}}) = \mathcal{L}(\mathcal{N})$

Observe that $|Q_{\mathcal{D}}| = 2^{|Q_{\mathcal{N}}|}$. Unfortunately, this exponential blow-up cannot be avoided.

A NFA $\mathcal{N}$ is complemented by:

1. NFA determinization $\mathcal{N} \Rightarrow \mathcal{D}_{\mathcal{N}}$
2. DFA complementation $\mathcal{D}_{\mathcal{N}} \Rightarrow \overline{\mathcal{D}_{\mathcal{N}}}$

Observe that the determinizing operation comes with an exponential blow-up in the size of the state-space of the automaton.

Reminder: due to determinization, this comes with an exponential blow-up.

Union construction

Take two NFAs $\mathcal{N}_1 = \langle Q_1, \Sigma, I_1, \delta_1, F_1 \rangle$ and $\mathcal{N}_2 = \langle Q_2, \Sigma, I_2, \delta_2, F_2 \rangle$ defined over the same alphabet Σ.
The union automaton $\mathcal{N}_1 \cup \mathcal{N}_2 = \langle Q, \Sigma, I, \delta, F \rangle$ is defined as:

$$Q = Q_1 \cup Q_2 \ ^a$$
$$I = I_1 \cup I_2$$
$$F = F_1 \cup F_2$$
$$\delta(q, \sigma) = \begin{cases} \delta_1(q, \sigma), & q \in Q_1 \\ \delta_2(q, \sigma), & q \in Q_2 \end{cases}$$

[a] We assume that Q_1 and Q_2 are disjoint.

Intuitively, $\mathcal{N}_1 \cup \mathcal{N}_2$ chooses nondeterministically to execute either $\mathcal{N}_1$ or $\mathcal{N}_2$.

$$\mathcal{L}(\mathcal{N}_1 \cup \mathcal{N}_2) = \mathcal{L}(\mathcal{N}_1) \cup \mathcal{L}(\mathcal{N}_2).$$

Synchronous product construction

Take two NFAs $\mathcal{N}_1 = \langle Q_1, \Sigma, I_1, \delta_1, F_1 \rangle$ and $\mathcal{N}_2 = \langle Q_2, \Sigma, I_2, \delta_2, F_2 \rangle$ defined over the same alphabet Σ.
The product automaton $\mathcal{N}_1 \times \mathcal{N}_2 = \langle Q, \Sigma, I, \delta, F \rangle$ is defined as:

$$Q = Q_1 \times Q_2$$
$$I = I_1 \times I_2$$
$$F = F_1 \times F_2$$
$$\delta((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))$$

Intuitively, $\mathcal{N}_1 \times \mathcal{N}_2$ executes $\mathcal{N}_1$ and $\mathcal{N}_2$ in parallel.

$$\mathcal{L}(\mathcal{N}_1 \times \mathcal{N}_2) = \mathcal{L}(\mathcal{N}_1) \cap \mathcal{L}(\mathcal{N}_2).$$

Some exercises

- $\mathcal{D}_1$ recognizes the words with an even number of b's
- $\mathcal{D}_2$ recognizes the words with at least an occurrence of a
- The product automaton $\mathcal{D}_1 \times \mathcal{D}_2$.

Regular expressions

$$\alpha := \varepsilon \mid a \mid \alpha \cdot \alpha \mid \alpha + \alpha \mid \alpha^*$$

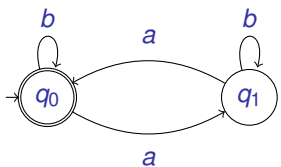
Every regular expression α denotes a language $\mathcal{L}(\alpha)$.

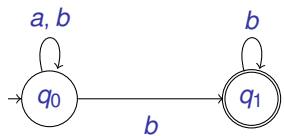
- The words ending with a b — $(a+b)^* \cdot b$
- The words with an a on every even index — $((a+b) \cdot a)^* \cdot (a+b+\epsilon)$
- The words with an odd number of a — $b^* \cdot (a \cdot b^*) \cdot ((a \cdot b^*) \cdot (a \cdot b^*))^*$

Theorem

1. For every regular expression α, there exists a NFA $\mathcal{N}_\alpha$ such that $\mathcal{L}(\alpha) = \mathcal{L}(\mathcal{N}_\alpha)$.
2. For every NFA $\mathcal{N}$, there exists a regular expression $\alpha_\mathcal{N}$ such that $\mathcal{L}(\alpha_\mathcal{N}) = \mathcal{L}(\mathcal{N})$.

Deterministic (DBA) and nondeterministic (NBA) Büchi automata are of the same type of DFA and NFA.



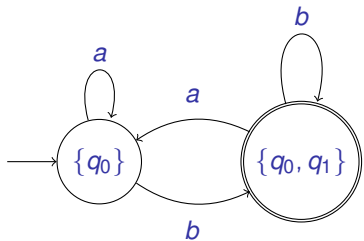


However, they read infinite words $w \in \Sigma^\omega$.

As there is no last state in the corresponding runs ρ, the acceptance condition is to visit a final state in F infinitely many times.

What are the languages recognized by the DBA and NBA depicted above?

Subset construction no longer works



What is the infinite word language accepted by this subset construction automaton?
The symbol b occurs infinitely many times (but also a might!) $(\Sigma^* \cdot b)^\omega$.

Theorem

The language $L = \{w \in \Sigma^\omega : w \text{ contains finitely many } a's\}$ can be recognized by a NBA but not by any DBA.

Corollary

NBAs are strictly more expressive than DBAs.

Theorem

For a given NBA $\mathcal{N}$, there exists a NBA $\overline{\mathcal{N}}$ such that $\mathcal{L}(\overline{\mathcal{N}}) = \Sigma^\omega \setminus \mathcal{L}(\mathcal{N})$.

However, the current techniques for the construction of $\overline{\mathcal{N}}$ are not trivial.

An entire research area in Formal Methods has been tackling this problem for many decades.

Luckily, we are not going to need this in our journey.
Just note that, as for NFAs, there is an unavoidable exponential blow-up.

Union construction

Take two NBAs $\mathcal{N}_1 = \langle Q_1, \Sigma, I_1, \delta_1, F_1 \rangle$ and $\mathcal{N}_2 = \langle Q_2, \Sigma, I_2, \delta_2, F_2 \rangle$ defined over the same alphabet Σ.
The union automaton $\mathcal{N}_1 \cup \mathcal{N}_2 = \langle Q, \Sigma, I, \delta, F \rangle$ is defined as:

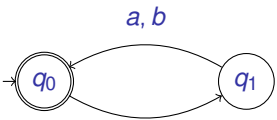
$$Q = Q_1 \cup Q_2$$
$$I = I_1 \cup I_2$$
$$F = F_1 \cup F_2$$
$$\delta(q, \sigma) = \begin{cases} \delta_1(q, \sigma), & q \in Q_1 \\ \delta_2(q, \sigma), & q \in Q_2 \end{cases}$$

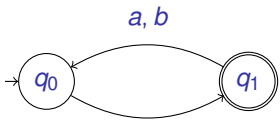
The union automaton guesses at the beginning whether to follow either $\mathcal{N}_1$ or $\mathcal{N}_2$.
Being on finite or infinite words does not make any difference.

$$\mathcal{L}(\mathcal{N}_1 \cup \mathcal{N}_2) = \mathcal{L}(\mathcal{N}_1) \cup \mathcal{L}(\mathcal{N}_2)$$

Closure properties

Intersection: synchronous product does not work





$$\mathcal{L}(\mathcal{D}_1 \times \mathcal{D}_2) = \emptyset!$$

We need a more clever way to deal with language intersection.

Another product construction

Take two NBAs $\mathcal{N}_1 = \langle Q_1, \Sigma, I_1, \delta_1, F_1 \rangle$ and $\mathcal{N}_2 = \langle Q_2, \Sigma, I_2, \delta_2, F_2 \rangle$ defined over the same alphabet Σ.

The product automaton $\mathcal{N}_1 \otimes \mathcal{N}_2 = \langle Q, \Sigma, I, \delta, F \rangle$ is defined as:

$$Q = Q_1 \times Q_2 \times \{1, 2\}$$

$$I = I_1 \times I_2 \times \{1\}$$

$$F = F_1 \times Q_2 \times \{1\}$$

$$\delta((q_1, q_2, 1), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 1), & \text{if } q_1 \notin F_1 \\ (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 2), & \text{if } q_1 \in F_1 \end{cases}$$

$$\delta((q_1, q_2, 2), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 2), & \text{if } q_2 \notin F_2 \\ (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma), 1), & \text{if } q_2 \in F_2 \end{cases}$$

$\mathcal{N}_1 \otimes \mathcal{N}_2$ switches the index every time a corresponding final state is hit.

$$\mathcal{L}(\mathcal{N}_1 \otimes \mathcal{N}_2) = \mathcal{L}(\mathcal{N}_1) \cap \mathcal{L}(\mathcal{N}_2).$$

A language is called ω-regular if it the union of expressions of the form $\alpha \cdot \beta^{\omega}$ with α and β being regular languages.

Theorem

1. For every ω-regular language L, there exists a NBA $\mathcal{N}_L$ such that $\mathcal{L}(\mathcal{N}) = L$.
2. For every NBA $\mathcal{N}$, the language $\mathcal{L}(\mathcal{N})$ is ω-regular.

Next class

- Nonemptiness problem of an NBA $\mathcal{N}$ $\mathcal{L}(\mathcal{N}) \neq \emptyset$
- From an LTL formula φ to an NBA $\mathcal{N}_\varphi$ $\mathcal{L}(\mathcal{N}_\varphi) = \{\pi \in \Sigma^\omega : \pi \models \varphi\}$
- Automata-Theoretic approach for LTL model-checking.