VI

AUTOMATA & FORMAL LANGUAGES

Sixth lecture: 18 October 2022

Fix an alphabet Σ

$$D = (\Sigma, Q, \delta, q_0, F)$$

is called a deterministic automaton

if:
- Q is a finite set — STATES
- $q_0 \in Q$ — START STATE
- $F \subseteq Q \setminus \{q_0\}$ — ACCEPT STATES
- $\delta : Q \times \Sigma \longrightarrow Q$ — TRANSITION FUNCTION

FROM LECTURE V

GRAMMARS
choice of the user; nonuniqueness

AUTOMATA
machine producing a unique computation sequence

FROM LECTURE V

UNIQUE COMPUTATION SEQUENCE

⇝ STATE SEQUENCE

Now formally:

Define by recursion

$$\hat{\delta} : Q \times W \longrightarrow Q$$

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

$$\hat{\delta}(q, wa) := \delta(\hat{\delta}(q, w), a)$$

$$\mathcal{L}(D) := \{ w ;\ \hat{\delta}(q_0, w) \in F \}$$

language accepted by D

Def. If $\mathcal{D} = (\Sigma, Q, \delta, q_0, F)$
$\mathcal{D}' = (\Sigma, Q', \delta', q_0', F')$
are two automata, then a map

$$f : Q \longrightarrow Q'$$

is called a homomorphism between $\mathcal{D}$ and $\mathcal{D}'$ if

(i) $\forall q \in Q \; \forall a \in \Sigma$
$$\delta'(f(q), a) = f(\delta(q, a))$$

(ii) $f(q_0) = q_0'$

(iii) $\forall q \in Q \quad q \in F$ if and only if $f(q) \in F'$.

Note: this was incorrect in the lecture. It only said "then".

By induction
$$\hat{\delta}'(f(q), w) = f(\hat{\delta}(q, w)) \quad (*)$$

f is called isomorphism if it is a bijective homomorphism

[If f is bij. homom. $\Rightarrow f^{-1}$ is homomorphism.]

REFLECTION ON FAILURE TO BE BIJECTIVE

Non-surjective: By (*), if f is not surjective, then no state that can be reached from q_0' can fail to be in the range of f.
All ACCESSIBLE states are in the range of f.

Non-injective: If $f(p) = f(q)$ then by (ii) they are either both accept or both reject states

Also $f(\hat{\delta}(q, w)) = \hat{\delta}'(f(q), w) = \hat{\delta}'(f(p), w) = f(\hat{\delta}(p, w))$

This property will be called INDISTINGUISHABLE

[For every w, $\hat{\delta}(q,w) \in F \leftrightarrow \hat{\delta}(p,w) \in F$]

So, we observe that failure to be surjective or injective only affect INACCESSIBLE states or pairs of INDISTINGUISHABLE state. These do not change the language accepted by D.

[More detail in §2.7.]

Proposition If $f: Q \longrightarrow Q'$ is a homomorphism from D to D', then $\mathcal{L}(D) = \mathcal{L}(D')$.

Proof.

$$
\begin{aligned}
w \in \mathcal{L}(D) &\overset{\text{Def}}{\iff} \hat{\delta}(q_0, w) \in F \\
&\overset{\text{(ii)}}{\iff} f(\hat{\delta}(q_0, w)) \in F' \\
&\overset{(*)}{\iff} \hat{\delta}'(f(q_0), w) \in F' \\
&\overset{\text{(i)}}{\iff} \hat{\delta}'(q_0', w) \in F' \\
&\overset{\text{Def}}{\iff} w \in \mathcal{L}(D').
\end{aligned}
$$

q.e.d.

Theorem 2.6 Any language of the form $\mathcal{L}(\mathcal{D})$ for a det. automaton $\mathcal{D}$ is regular.

Proof. $\mathcal{D} = (\Sigma, Q, \delta, q_0, F)$

Define $G := (\Sigma, Q, P, q_0)$ with

$$P := \{ p \to aq \;;\; \boxed{\delta(p,a) = q} \}$$
$$\cup \{ p \to a \;;\; \boxed{\delta(p,a) \in F} \}$$

Claim $\mathcal{L}(\mathcal{D}) = \mathcal{L}(G)$.

Fix $w = a_0 \dots a_n$.

$$w \in \mathcal{L}(\mathcal{D}) \overset{\text{Def}}{\iff} \hat{\delta}(q_0, w) \in F$$

$$\overset{\text{Def of } \hat{\delta}}{\iff} \exists (q_0, \dots, q_{n+1}) \text{ s.t. } \boxed{q_{i+1} = \delta(q_i, a_i)} \text{ and } \boxed{q_{n+1} \in F}.$$

$$\overset{\text{Def of } G}{\iff} \exists (q_0, \dots, q_{n+1}) \text{ s.t. } q_i \to a_i q_{i+1} \in P \;\&\; q_n \to a_n \in P$$

$$\iff \exists (q_0, \dots, q_{n+1}) \text{ s.t. } q_0 \xrightarrow{G}_1 a_0 q_1 \xrightarrow{G}_1 \dots \xrightarrow{G}_1 a_0 \dots a_n q_n \xrightarrow{G}_1 w$$

$$\overset{\text{by L 2.1}}{\iff} w \in \mathcal{L}(G)$$

q.e.d.

Goal: Prove the converse:
i.e., if L is regular, find det. automaton D s.t. $L = \mathcal{L}(D)$.

Idea: Reverse the operation, i.e.,
if you see

$$p \longrightarrow aq,$$

define $\delta(p,a) = q$.

Alas, it is possible that both

$$p \longrightarrow aq \text{ and}$$
$$p \longrightarrow aq' \text{ with } q \neq q'$$

are in P.
So δ would not be a function!

Thus: the naive idea doesn't work.

§ 2.3 Nondeterministic automata

$$N = (\Sigma, Q, \delta, q_0, F)$$

is called a nondeterministic automaton

if Q finite set of states

$q_0 \in Q$

$F \subseteq Q \setminus \{q_0\}$

the same as in deterministic automata

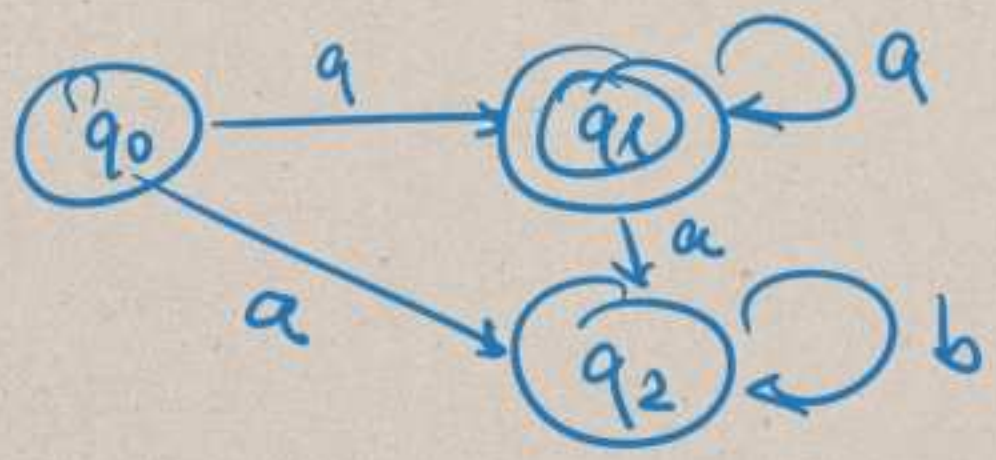
$$\delta : Q \times \Sigma \longrightarrow \mathcal{P}(Q)$$

INTERPRETATION:

$\delta(q,a)$ is the set of states that possible next moves of the nondet. automaton.

Graphical representation

Exactly the same except that a node can have multiple or no arrows with the same letter



Analogue of $\hat{\delta}$ for nondeterministic automata:

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

$$\hat{\delta}(q, wa) := \bigcup_{p \in \hat{\delta}(q,w)} \delta(p, a)$$

This produces a unique state set sequence

We define

$$\mathcal{L}(N) := \{w ; \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$$

Remark Clearly, deterministic automata can be seen as a special case of nondeterministic autom.

In general, nondet. aut. seem to provide much more freedom.

This is not the case:

Theorem 2.7 For every nondet. aut. N there is a det. aut. D s.t.

$$\mathcal{L}(N) = \mathcal{L}(D).$$

Proof. "subset construction".

Suppose $N = (\Sigma, Q, \delta, q_0, F)$ is given.

Define $D = (\Sigma, \mathcal{P}(Q), \Delta, \{q_0\}, G)$

by $$\Delta(X, a) := \bigcup_{q \in X} \delta(q, a)$$

$$X \in G :\Longleftrightarrow X \cap F \neq \emptyset.$$

Claim: $\mathcal{L}(N) = \mathcal{L}(D)$.

Consider the state sequence of D on input $w = a_0 \dots a_n$:

$$\boxed{\begin{aligned} X_0 &= \{q_0\} \\ X_{i+1} &= \bigcup_{q \in X_i} \delta(q, a_i) \end{aligned}} \Longrightarrow X_i = Y_i \quad (\star)$$

[induction]

and the state set seq. of N on input w

$$\boxed{\begin{aligned} Y_0 &= \{q_0\} \\ Y_{i+1} &= \bigcup_{q \in Y_i} \delta(q, a_i) \end{aligned}}$$

Proof of Claim:

$$w \in \mathcal{L}(D) \overset{\text{Def of } \mathcal{L}(D)}{\Longleftrightarrow} X_{u+1} \in G$$

$$\overset{\text{Def of } G}{\Longleftrightarrow} X_{u+1} \cap F \neq \emptyset$$

$$\Longleftrightarrow Y_{u+1} \cap F \neq \emptyset$$

$$\Longleftrightarrow w \in \mathcal{L}(N). \quad \text{q.e.d.}$$

(A)

Remark Transforming a nondet. auto into a det. automaton costs a high price: the automaton D has 2^u states if N had u states.

Theorem If G is a regular grammar, then there is a nondet. automaton N s.t. $\mathcal{L}(G) = \mathcal{L}(N)$.

Proof. $G = (\Sigma, V, P, S)$

Let $H \notin \Sigma \cup V$ "halt state"

$Q := V \cup \{H\}$ $N := (\Sigma, Q, \delta, S, \{H\})$ with

$$\delta(A,a) := \begin{cases} \{B; A \to aB \in P\} & \text{if } A \to a \notin P \\ \{B; A \to aB \in P\} \cup \{H\} & \text{if } A \to a \in P \end{cases}$$

Claim: $\mathcal{L}(G) = \mathcal{L}(N)$.

$$w \in \mathcal{L}(G) \overset{L2.1}{\iff} \exists (A_0, \dots, A_{n+1})$$

$$S = A_0 \xrightarrow[G]{}_1 a_0 A_1 \xrightarrow[G]{}_1 \dots \xrightarrow[G]{}_1 a_0 \dots a_{n-1} A_n \xrightarrow[G]{}_1 w$$

is a G-derivation

$$\iff \exists (A_0, \dots, A_{n+1})$$

$$S = A_0 \;\&\; A_i \longrightarrow a_i A_{i+1} \in P \;\&\; A_n \longrightarrow a_n \in P$$

$$\iff \exists (A_0, \dots, A_{n+1})$$

$$S = A_0 \;\&\; A_{i+1} \in \delta(A_i, a_i) \;\&\; H \in \delta(A_n, a_n)$$

$$\iff H \in \hat{\delta}(S, w)$$

$$\iff \hat{\delta}(S, w) \cap F = \hat{\delta}(S, w) \cap \{H\} \neq \emptyset$$

$$\iff w \in \mathcal{L}(N)$$

q.e.d.