V

AUTOMATA & FORMAL LANGUAGES
FIFTH LECTURE

15 October 2022

RECAP

Chapter 1: Languages & Grammars

Grammar $G = (\Sigma, V, P, S)$

Its language $\mathcal{L}(G)$

COUNTABLY MANY

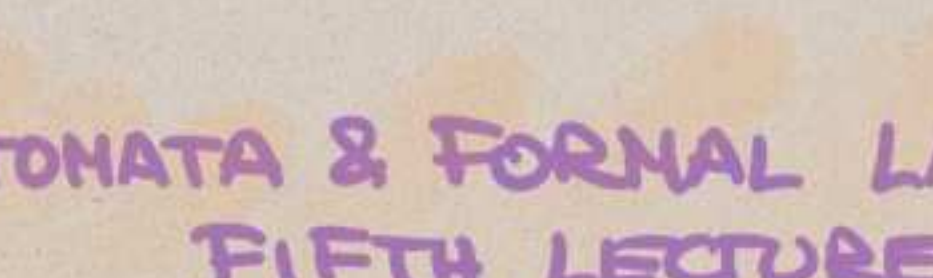
Languages $L \subseteq \mathbb{W} = \Sigma^*$

UNCOUNTABLY MANY

Chomsky hierarchy

all languages

Figure 1: The Chomsky hierarchy



0 : any grammar

1 : context-sensitive grammar

[$\Leftrightarrow$ noncontracting]

Example Sheet #1 Example 6

2 : context-free

3 : regular

FURTHER RECAP

Two important properties of classes of languages:

(1) Solvability of decision problems
- WORD PROBLEM $w \in L(G)$
- EMPTINESS PROBLEM $L(G) = \emptyset$
- EQUIVALENCE PROBLEM $L(G) = L(G')$

Theorem: The word problem for noncontracting grammars is solvable.

(2) Closure properties
- Concatenation
- Union
- Intersection
- Complement
- Difference

Theorem: Types 1,2,3 closed under union
Types 1,2 closed under concatenation

CHAPTER 2: REGULAR LANGUAGES

RECAP

G is regular if every production rule is of the form

TERMINAL RULE ⟶ $A \to a$ or

NONTERMINAL RULE ⟶ $A \to aB$

for $A, B \in V$ and $a \in \Sigma$.

§2.1 Understanding regular derivations

Let G be a regular grammar.

Lemma 2.1a If $S \xrightarrow{G} \alpha$, then $\alpha \in W \cup WV$.

Proof. By ind. on length of derivation.

Length zero: $\alpha = S = \varepsilon S \in WV$. ✓

Suppose works for length n and show for $n+1$:

$$S \xrightarrow{G} \beta \xrightarrow[1]{G} \alpha$$

length n

By IH: $\beta \in W \cup WV$.

Case 1 $\beta \in W$, then there is no variable and so rule can be applied
Contradiction!

Case 2 $\beta = wA$ for some $w \in W$ & $A \in V$.

So we apply either a rule $A \to a$ (2a)
$A \to aB$ (2b).

Subcase 2a $\alpha = wa \in W$

Subcase 2b $\alpha = waB \in WV$.

q.e.d.

Lemma 2.1b If $S \xrightarrow{G} w$, then that derivation has length $|w|$ and consists of precisely $|w|-1$ nonterminal and one (final) terminal rule application.

Pf. Terminal rules:
- keep length the same
- reduce # variables by one

Nonterminal rules:
- increase length by one
- keep # variables the same

So clearly a derivation $S \xrightarrow{G} w$ must have $|w|-1$ nonterminal rules applied.

By L2.1a the # of variables is always either zero or one.

Therefore: we have one terminal rule application and it must be at the end.

q.e.d.

Note Derivation still need not be unique.

Application

An alternative construction for concatenation grammars:

$G = (\Sigma, V, P, S)$ w.l.o.g. $V \cap V' = \emptyset$

$G' = (\Sigma, V', P', S')$

Define $P^* := \{ A \to aB ;\ A \to aB \in P \}$

$\cup \{ A \to aS' ;\ A \to a \in P \}$

$H := (\Sigma, V \cup V', P^* \cup P', S)$

Claim $\mathcal{L}(H) = \mathcal{L}(G)\mathcal{L}(G')$

Suppose $S \xrightarrow{G} v$ and $S' \xrightarrow{G'} w$

"$\supseteq$".

$S \xrightarrow{H} vS'$ and $vS' \xrightarrow{G'} vw$

$vS' \xrightarrow{H} vw$

$S \xrightarrow{H} vw$

"$\subseteq$" Suppose $S \xrightarrow{H} u$.

Then $S = \sigma_0 \xrightarrow{H}_1 \cdots \xrightarrow{H}_1 \sigma_n = u$ with $\sigma_i = wX$ $i < n$

By L2.1b

We have that an initial segment of the σ_i are in V until rewritten as S' and then the σ_i are in V'.

Therefore there is some $v, w \in \mathbb{N}$ s.t.

$$u = vw$$

and $S \xrightarrow{G} v$, $S \xrightarrow{H} vS'$.)

$$vS' \xrightarrow{H} u = vw$$

$$\Longrightarrow S' \xrightarrow{G'} w.$$

Thus $u \in \mathcal{L}(G)\mathcal{L}(G')$.

COROLLARY

Regular languages are closed under concatenation.

§2.2 Deterministic automata

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

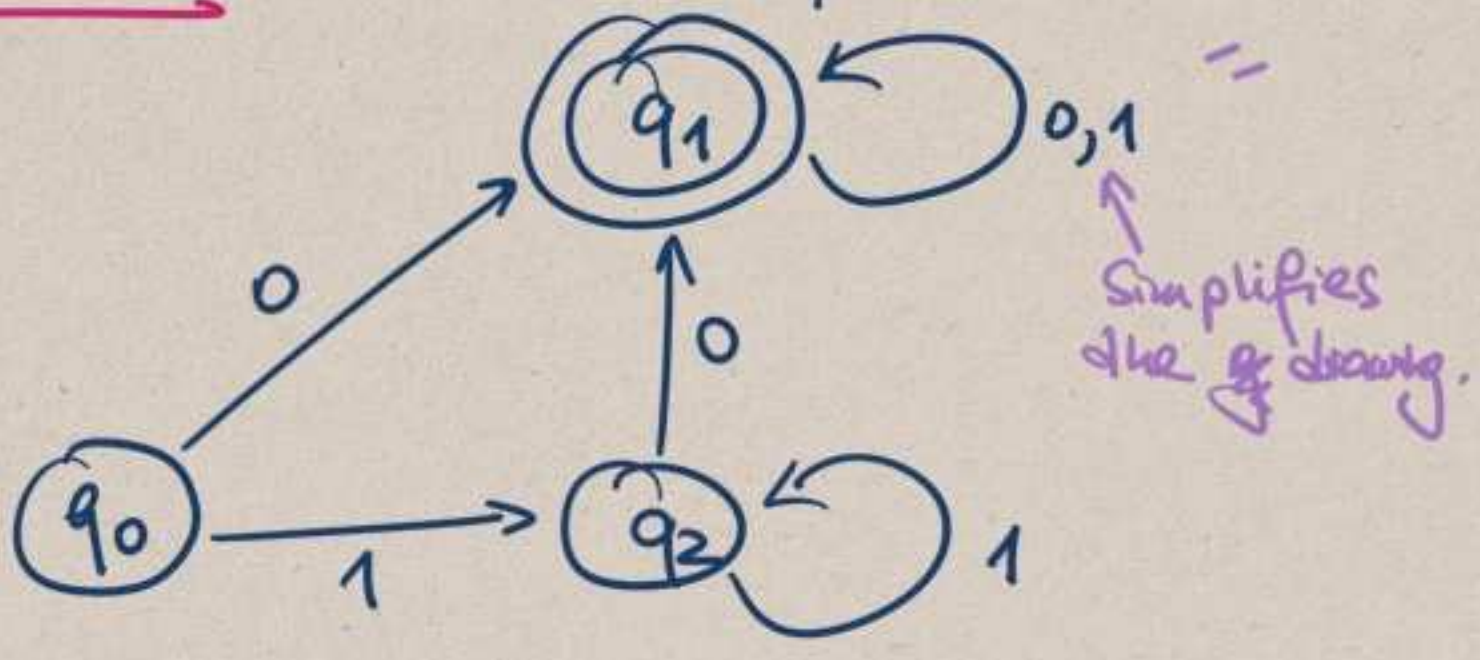
Graphical representation by **labelled directed graphs**

- Nodes/vertices: elements of Q
- Nodes get single circle: $q \notin F$
  q double circle: $q \in F$
- Each node has $|\Sigma|$ outgoing arrows labelled by letters.

$\Sigma = \{0,1\}$ $Q = \{q_0, q_1, q_2\}$

$F = \{q_1\}$

EXAMPLE



Simplifies the drawing.

INTERPRETATION

1. We think of the machine being in state q_0 at the beginning.
2. We feed D a word w and it reads the word symbol by symbol.
3. When it reads a letter a, it takes its current state and moves to state $\delta(q,a)$.
4. Once done, it is in some state q. It ACCEPTS w, if $q \in F$; REJECTS w, otherwise.

Now formally:

Define by recursion

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

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

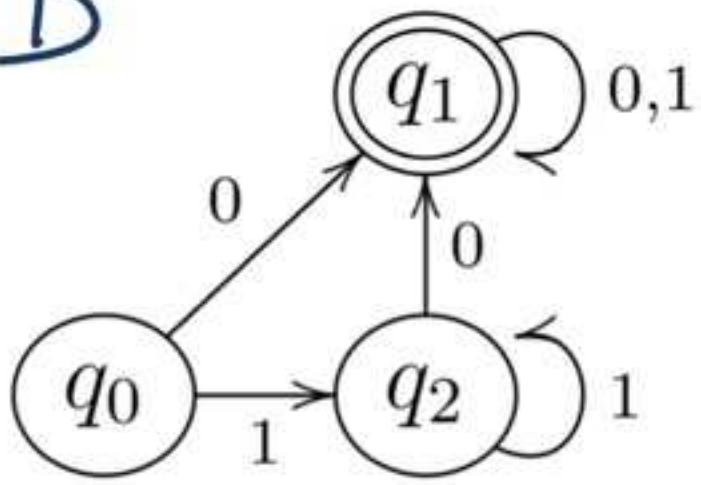
$$\hat{\delta}(q, wa) := \delta(\hat{\delta}(q, w), a)$$

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

language accepted by D

The sequence of states produced from q_0 at reading w is uniquely determined (length $|w|+1$) and called the state sequence of the computation.

D



Claim

$\mathcal{L}(D) =$
$\{w;$ w contains at least one $0\}$

Which word w will end up in which state?

q_0: Since q_0 has no incoming nodes
$\hat{\delta}(q_0, w) = q_0 \iff w = \varepsilon.$

q_1: Since both transitions lead from q_1 to q_1,
you can never leave q_1;
since all 0-transitions lead to q_1,
every ~~word~~ string w/ a zero will
be leading to q_1.

q_2: All strings in $\{1\}^+$ will end up in q_2.

Since q_1 is the only accepting state,
this proves the claim.

Goal for lecture VI:

Show that languages accepted by an automaton are regular.

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

$$f : Q \longrightarrow Q'$$

is called a homomorphism between D and 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".

The last page was modified after the lecture:

Note that if f is a bijection, then f^{-1} is also a homeomorphism.
A bijective homomorphism is called isomorphism.

Proposition 2.5

If there is a homeomorphism from D to D', then

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

[Remark. Note that any homomorphism, even if not bijective will hit every state of D that matters for $\mathcal{L}(D')$.

Also, if f is not injective, i.e., $f(p) = f(q)$ for some $p \neq q$, then p and q cannot really differ in any way that is relevant for $\mathcal{L}(D)$.]