AUTOMATA & FORMAL LANGUAGES

Lecture III

11 October 2023

NOTATION If $R = (\Omega, P)$ is a rewrite system, $\sigma, \tau \in \Omega^*$, then we write

$$\sigma \xrightarrow{R}_1 \tau$$

R rewrites σ in one step as τ

$$:\Longleftrightarrow$$

$\exists \alpha, \beta, \gamma, \delta \in \Omega^*$ s.t.

$\sigma = \alpha\beta\gamma$,

$\tau = \alpha\delta\gamma$,

$\beta \longrightarrow \delta \in P$.

We let the relation $\xrightarrow{R}$ be the transitive and reflexive closure of $\xrightarrow{R}_1$.

This means $\sigma \xrightarrow{R} \tau$ iff

(1) $\sigma = \tau$

(2) there is a sequence of strings $\sigma_0, \dots, \sigma_n$ s.t. $\sigma = \sigma_0$, $\tau = \sigma_n$ and $\sigma_k \xrightarrow{R}_1 \sigma_{k+1}$.

a R-derivation of τ from σ

This is called derivation of length n.

Note that a derivation of length n is a sequence of length $n+1$.

R derives τ from σ.

R rewrites σ as τ

R produces τ from σ

$\mathcal{D}(R, \alpha) := \{\beta ; \alpha \xrightarrow{R} \beta\}$

Recap

pp. 10–11 of Lecture II

$$\mathcal{D}(G, \alpha) := \{\beta ; \alpha \xrightarrow{G} \beta\}$$

$$\mathbb{W} := \Sigma^*$$

$$\mathcal{L}(G) := \mathbb{W} \cap \mathcal{D}(G, S)$$

GRAMMARS

Def. $G = (\Sigma, V, P, S)$ is called a (formal) grammar over Σ if

1. $\Sigma \cap V = \emptyset$
2. $\Omega := \Sigma \cup V$ is finite nonempty set of symbols

terminal symbols or letters

nonterminal symbols or variables

Σ is called the alphabet

3. (Ω, P) is a rewrite system
4. $S \in V$ called the start symbol

We call $\mathbb{W} := \Sigma^*$ the set of WORDS.

The definitions for rewrite systems still make sense:

$\alpha \xrightarrow{G}_1 \beta$, $\mathcal{D}(G, \alpha)$

$\alpha \xrightarrow{G} \beta$

$$\mathcal{L}(G) := \mathbb{W} \cap \mathcal{D}(G, S)$$

LANGUAGE generated by G.

Example

From Lecture II, page 12.

Example $G_0 = (\Sigma, V, P_0, S)$

$$\Sigma := \{a\}$$
$$V := \{S\}$$
$$P_0 := \{S \longrightarrow aaS,\ S \longrightarrow a\}$$

Claim $\mathcal{L}(G_0) = \{a^{2n+1}\,;\ n \in \mathbb{N}\}$

Easy to see that $\supseteq$;
to see $\subseteq$, observe that every derivable string has odd length.

[Details will be repeated at the beginning of Lecture II.]

Proof of Claim:

"$\supseteq$":

$$S \xrightarrow{G}_1 aaS \xrightarrow{G}_1 \cdots \xrightarrow{G}_1 (aa)^n S = a^{2n}S \xrightarrow{G}_1 a^{2n}a = a^{2n+1}$$

"$\subseteq$": I claim that if $\alpha \in \mathcal{D}(G, S)$, then $|\alpha|$ is odd.

Proof by induction by length of derivation:
Only string w/ derivation of length 0 is S and $|S| = 1$.
Both of our rules preserve parity. q.e.d.

A proof that $\alpha \notin \mathcal{D}(G, S)$ is always an induction proof of an invariant property shared by all elts of $\mathcal{D}(G, S)$, but not α.

An analysis of the argument in Example 1.9 shows that if in a grammar all production rules preserve oddness of length and we can provide a derivation of a^{2n+1}, then the grammar will produce the same language. E.g., $G_i = (\{a\}, \{S\}, P_i, S)$ with

$$\begin{aligned}
P_1 &:= \{S \to aSa, S \to a\},\\
P_2 &:= \{S \to Saa, S \to a\},\\
P_3 &:= \{S \to aaS, S \to aaSaa, S \to a\},\\
P_4 &:= \{S \to aaS, S \to Saa, S \to aSa, S \to a\},\\
P_5 &:= \{S \to aaS, aSa \to aaa, S \to a\},\\
P_6 &:= \{S \to aaS, aaS \to aSa, S \to a\}, \text{ or}\\
P_7 &:= \{S \to aaS, aaS \to a, S \to a\}, \text{ etc.}
\end{aligned}$$

These grammars all have the properties

(1) $\{a^{2n+1}; n \in \mathbb{N}\} \subseteq \mathcal{L}(G_i)$

(2) all rules preserve parity

Thus: $\mathcal{L}(G_i) = \{a^{2n+1}; n \in \mathbb{N}\}$

<u>Definition</u> G, G' are <u>equivalent</u> if $\mathcal{L}(G) = \mathcal{L}(G')$.

Observe: $G_0, G_1, \ldots, G_7$ are equivalent.

But these are not entirely interchangeable. E.g.

$\mathcal{D}(G_0, S) = \{a^{2n+1}; n \in \mathbb{N}\} \cup \{a^{2n}S; n \in \mathbb{N}\}$

$\mathcal{D}(G_1, S) = \{a^{2n+1}; n \in \mathbb{N}\} \cup \{a^{n}Sa^{n}; n \in \mathbb{N}\}$

Definition 1.10. Let Σ be an alphabet and let $G = (\Sigma, V, P, S)$ and $G' = (\Sigma, V', P', S')$ be two grammars over Σ. Let $f : \Omega \to \Omega'$ be any function and extend it by recursion to Ω^*. We say that f is an *isomorphism between G and G'* if

(i) it is the identity on Σ, i.e., $f(a) = a$ for all $a \in \Sigma$;

(ii) $f(S) = S'$;

(iii) the restriction $f{\upharpoonright}V$ is a bijection between V and V'; and

(iv) for each $\alpha, \beta \in \Omega^*$, we have $\alpha \to \beta \in P$ if and only if $f(\alpha) \to f(\beta) \in P'$.

If there is an isomorphism between G and G', we also say that the two grammars are *isomorphic*.

Remember that, $f : \Omega \to \Omega'$
it lifts to the Ω-strings.

Prop. Isomorphic grammars are equivalent.

Proof If f is an iso from G to G', [by (iii)]
then f^{-1} is an iso from G' to G.
So, it's enough to show $\mathcal{L}(G) \subseteq \mathcal{L}(G')$.
If $\sigma_0, \dots, \sigma_n$ is a G-derivation of w from S

$\Downarrow f$ $\Downarrow f$

$f(\sigma_0), \dots, f(\sigma_n)$ is a G'-derivation of $f(w)$ from $f(S)$ [by (iv)]

$f(w) = w$ [by (i)] $f(S) = S'$ [by (ii)]

q.e.d.

<u>Prop.</u> / <u>Observation</u> Let Σ be fixed and V, V' two sets of variables with $|V| = |V'|$, say with bijection $f: V \longrightarrow V'$.

Let $G = (\Sigma, V, P, S)$ be a grammar.

Define $S' := f(S)$

$P' := \{ f(\alpha) \longrightarrow f(\beta) ; \alpha \to \beta \in P \}$.

Then $G' := (\Sigma, V', P', S')$ is isomorphic to G.

[Obvious!]

<u>Remark</u>: (a) This means that the set of variables is irrelevant, only its size matters.

(b) It also means that "up to equivalence" we can always assume that grammars have disjoint sets of variables.

Theorem For every fixed Σ, there are only countably many languages $L \subseteq \mathbb{W}$ s.t. there is a G grammar w/ $\mathcal{L}(G) = L$.

Proof. For any fixed V, there is a surjection onto

$$\mathcal{L}_V := \{L \subseteq \mathbb{W} \,;\, \exists\, G = (\Sigma, V, P, S) \text{ s.t. } \mathcal{L}(G) = L\}$$

from the set of rewrite systems on $\Omega = \Sigma \cup V$.

We proved that this is ctble, and so is $\mathcal{L}_V$.

If $\mathcal{L}_n := \{L \subseteq \mathbb{W} \,;\, \exists\, G = (\Sigma, V, P, S) \text{ with } |V| = n \text{ and } \mathcal{L}(G) = L\}$

Observation says: if $|V| = n$, then $\mathcal{L}_V = \mathcal{L}_n$.

So $\{L \subseteq \mathbb{W} \,;\, \exists\, G \text{ with } \mathcal{L}(G) = L\}$

$$= \bigcup_{n \in \mathbb{N}} \mathcal{L}_n .$$

So, it's countable as a ctble union of ctble sets.

q.e.d.

The CHOMSKY hierarchy

Let $\alpha \longrightarrow \beta$ be a production rule.

noncontracting (nc): if $|\alpha| \leq |\beta|$

context-sensitive (c-s): if $\alpha = \gamma A \delta$, $\beta = \gamma \eta \delta$ with $\gamma, \delta, \eta \in \Omega^*$, $A \in V$, $|\eta| \geq 1$

context-free (c-f): if $\alpha = A$, $A \in V$, $|\beta| \geq 1$

regular (reg): if $\alpha = A$ and $\begin{pmatrix} \beta = a \text{ or} \\ \beta = aB \end{pmatrix}$ for $A, B \in V$, $a \in \Sigma$

A grammar is nc, c-s, c-f, reg if all of its production rules are.

A language L is nc, c-s, c-f, reg if there is a nc, c-s, c-f, reg grammar G s.t. $\mathcal{L}(G) = L$, respectively.

<u>Remark</u>: (a) reg $\Longrightarrow$ c-f $\Longrightarrow$ c-s $\Longrightarrow$ nc.

This implication holds for rules, grammars, & languages.

(b) Clearly, there counterexamples for the reverse directions of (a) for rules & grammars.

But that does not mean that we have ctrex for languages.

Conceivable:

G, G' equivalent but

G is regular and

G' is not even noncontracting.

Then $\mathcal{L}(G') = \mathcal{L}(G)$ is regular.

(c) If G is noncontracting and

$\alpha \in \mathcal{D}(G, S)$,

then $|\alpha| \geq 1$.

In particular, $\varepsilon \notin \mathcal{L}(G)$.

Therefore, we mostly work on

$$\mathbb{W}^+ := \mathbb{W} \setminus \{\varepsilon\}.$$

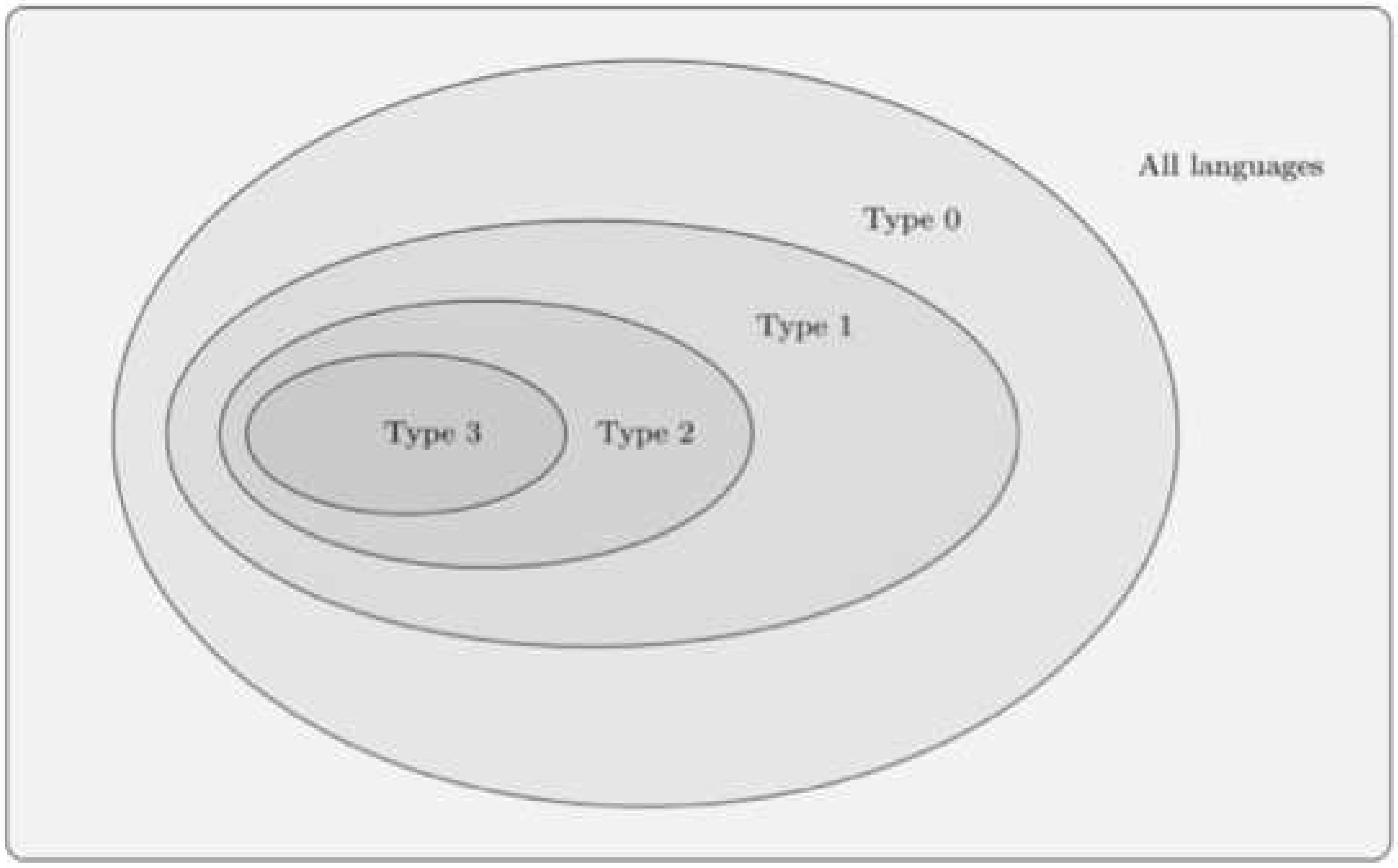

Figure 1: The Chomsky hierarchy

Chomsky defined:

L is type 0 if there is G s.t.
$L = \mathcal{L}(G)$.

L is type 1 if it is context-sensitive
—"— 2 ______ context-free
—"— 3 ______ regular.

Observe ① Noncontracting languages are missing. Why?

② Question: Is the diagram PROPER, i.e., is each area in the picture populated by example.