1 Chomsky Hierarchy

Grammars for each task

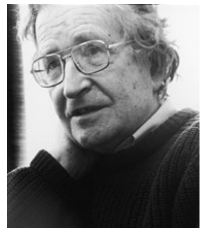


Figure 1: Noam Chomsky

- Different types of rules, allow one to describe different aspects of natural language
- These grammars form a hierarchy

Grammars in General

All grammars we consider will be of the form $G = (V, \Sigma, R, S)$

- V is a finite set of variables
- Σ is a finite set of terminals
- R is a finite set of rules
- S is the start symbol

The different grammars will be determined by the form of the rules in R.

1.1 Regular Languages

Type 3 Grammars

The rules in a type 3 grammar are of the form

$$A \to aB \qquad \text{or} \qquad A \to a$$

where $A, B \in V$ and $a \in \Sigma \cup \{\epsilon\}$.

We say $\alpha A \beta \Rightarrow_G \alpha\gamma\beta$ iff $A \to \gamma \in R$. $\mathbf{L}(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow}_G w\}$

1.1.1 Type 3 Grammars and Regularity

Type 3 Grammars and Regularity

Proposition 1. *If G is Type 3 grammar then $\mathbf{L}(G)$ is regular. Conversely, if L is regular then there is a Type 3 grammar G such that $L = \mathbf{L}(G)$.*

Proof. Let $G = (V, \Sigma, R, S)$ be a type 3 grammar. Consider the NFA $M = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = V \cup \{q_F\}$, where $q_F \notin V$
- $q_0 = S$
- $F = \{q_F\}$
- $\delta(A, a) = \{B \mid \text{if } A \to aB \in R\} \cup \{q_F \mid \text{if } A \to a \in R\}$ for $A \in V$. And $\delta(q_F, a) = \emptyset$ for all a.

$\mathbf{L}(M) = L(G)$ as $\forall A \in V$, $\forall w \in \Sigma^*$, $A \overset{*}{\Rightarrow}_G w$ iff $A \xrightarrow{w}_M q_F$.

Conversely, let $M = (Q, \Sigma, \delta, q_0, F)$ be a NFA recognizing L. Consider $G = (V, \Sigma, R, S)$ where

- $V = Q$
- $S = q_0$
- $q_1 \to aq_2 \in R$ iff $q_2 \in \delta(q_1, a)$ and $q \to \epsilon \in R$ iff $q \in F$.

We can show, for any $q, q' \in Q$ and $w \in \Sigma^*$, $q \xrightarrow{w}_M q'$ iff $q \overset{*}{\Rightarrow}_G wq'$. Thus, $\mathbf{L}(M) = \mathbf{L}(G)$. □

1.2 Context-free Languages

Type 2 Grammars

The rules in a type 2 grammar are of the form

$$A \to \beta$$

where $A \in V$ and $\beta \in (\Sigma \cup V)^*$.

We say $\alpha A \beta \Rightarrow_G \alpha\gamma\beta$ iff $A \to \gamma \in R$. $\mathbf{L}(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow}_G w\}$

By definition, Type 2 grammars describe exactly the class of context-free languages.

1.3 Beyond Context-Free Languages

1.3.1 Type 0 Grammars

Type 0 Grammars

The rules in a type 0 grammar are of the form

$$\alpha \to \beta$$

where $\alpha, \beta \in (\Sigma \cup V)^*$.

We say $\gamma_1 \alpha \gamma_2 \Rightarrow_G \gamma_1 \beta \gamma_2$ iff $\alpha \to \beta \in R$. $\mathbf{L}(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow}_G w\}$

Example of Type 0 Grammar

Example 2. Consider the grammar G with $\Sigma = \{a\}$ with

$$\begin{array}{lll} S \to \$Ca\# \mid a \mid \epsilon & Ca \to aaC & \$D \to \$C \\ C\# \to D\# \mid E & aD \to Da & aE \to Ea \\ \$E \to \epsilon & & \end{array}$$

The following are derivations in this grammar

$$S \Rightarrow \$Ca\# \Rightarrow \$aaC\# \Rightarrow \$aaE \Rightarrow \$aEa \Rightarrow \$Eaa \Rightarrow aa$$

$$\begin{aligned} S &\Rightarrow \$Ca\# \Rightarrow \$aaC\# \Rightarrow \$aaD\# \Rightarrow \$aDa\# \Rightarrow \$Daa\# \Rightarrow \$Caa\# \\ &\Rightarrow \$aaCa\# \Rightarrow \$aaaaC\# \Rightarrow \$aaaaE \Rightarrow \$aaaEa \Rightarrow \$aaEaa \\ &\Rightarrow \$aEaaa \Rightarrow \$Eaaaa \Rightarrow aaaa \end{aligned}$$

$\mathbf{L}(G) = \{a^i \mid i \text{ is a power of } 2\}$

Expressive Power of Type 0 Grammars

Recall that any decision problem can be thought of as a formal language L, where $x \in L$ iff the answer on input x is "yes".

Proposition 3. *A decision problem L can be "solved on computers" iff L can be described by a Type 0 grammar.*

Proof. Need to develop some theory, that we will see in the next few weeks. □

1.3.2 Type 1 Grammars

Type 1 Grammars

The rules in a type 1 grammar are of the form

$$\alpha \to \beta$$

where $\alpha, \beta \in (\Sigma \cup V)^*$ *and* $|\alpha| \leq |\beta|$.

We say $\gamma_1 \alpha \gamma_2 \Rightarrow_G \gamma_1 \beta \gamma_2$ iff $\alpha \to \beta \in R$. $\mathbf{L}(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow}_G w\}$

Normal Form for Type 1 Grammars

We can define a normal form for Type 1 grammars where all rules are of the form

$$\alpha_1 A \alpha_2 \to \alpha_1 \beta \alpha_2$$

Thus, the rules in Type 1, can be seen as rules of a CFG where a variable A is replaced by a string β in one step, with the only difference being that rule can be applied only in the context $\alpha_1 \Box \alpha_2$.

Thus, languages described by Type 1 grammars are called *context-sensitive languages.*

1.3.3 Hierarchy

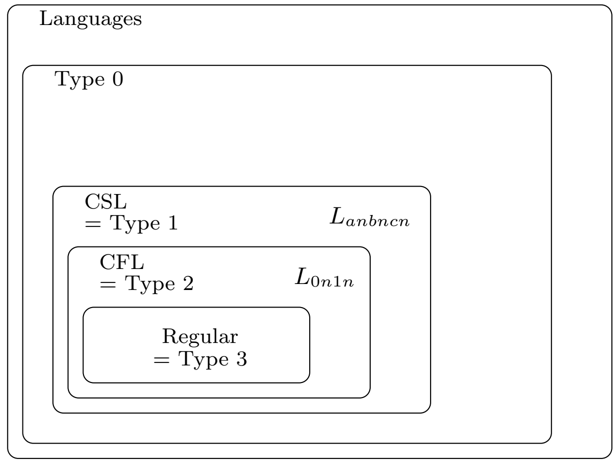
Chomsky Hierarchy

Theorem 4. *Type 0, Type 1, Type 2, and Type 3 grammars define a* strict *hierarchy of formal languages.*

Proof. Clearly a Type 3 grammar is a special Type 2 grammar, a Type 2 grammar is a special Type 1 grammar, and a Type 1 grammar is special Type 0 grammar.

Moreover, there is a language that has a Type 2 grammar but no Type 3 grammar ($L = \{0^n 1^n \mid n \geq 0\}$), a language that has a Type 1 grammar but no Type 2 grammar ($L = \{a^n b^n c^n \mid n \geq 0\}$), and a language with a Type 0 grammar but no Type 1 grammar. $\Box$

Overview of Languages


Languages
Type 0
CSL
= Type 1
L_{anbncn}
CFL
= Type 2
L_{0n1n}
Regular
= Type 3
