Context-free Grammars, Regular Grammars and Chomsky's Hierarchy

A *context-free grammar* G over alfabet Σ needs a finite set V of *auxiliary* (help) symbols and consists of *production rules* of the form

$$X \to w$$

with $X \in V$ and $w \in (\Sigma \cup V)^*$. There is an $S \in V$ (*start*)

Using G a language is generated using a relation $\Rightarrow$ ('*produces*') defined as follows

(u, v, w, x, y are arbitrary elements of $(\Sigma \cup V)^*$)

$$X \to w \quad \text{implies} \quad xXy \Rightarrow xwy$$
$$u \Rightarrow v,\ v \Rightarrow w \quad \text{implies} \quad u \Rightarrow w$$

The *language generated by G* is

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow w\}$$

Let G be given by

$$V = \{S, A, B\}$$
$$\Sigma = \{a, b\}$$

S	$\to$	aB | bA
A	$\to$	a| aS | bAA
B	$\to$	b| bS | aBB

Show that $ababba \in L(G)$ but $baaababba \notin L(G)$

Characterize $L(G)$; give motivation.

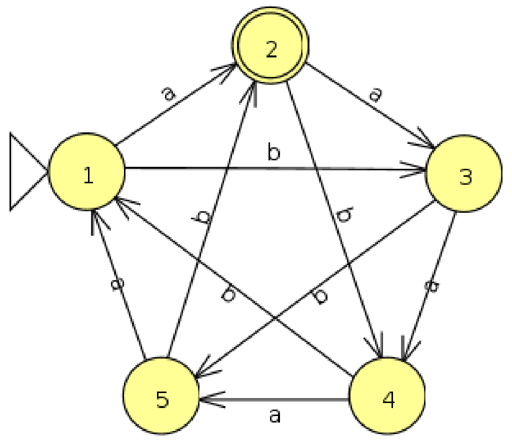
Suppose L is regular . . . is it context-free?

We have a choice of

- taking e such that $L = L(e)$ and transforming the regular expression e to a grammar, or
- taking DFA M such that $L = L(M)$ and transforming the DFA M to a grammar, or
- taking NFA M such that $L = L(M)$ and transforming the NFA M to a grammar, or
- taking NFA$_\lambda$ M such that $L = L(M)$ and transforming the NFA$_\lambda$ M to a grammar

. . . We start from a DFA!

Example. Given a DFA



Can we find a regular grammar generating the same language?

Yes:

S=1	$\rightarrow$	a2 | b3
2	$\rightarrow$	a3 | b4 | λ
3	$\rightarrow$	a4 | b5
4	$\rightarrow$	a5 | b1
5	$\rightarrow$	a1 | b2

This works in general: every DFA M can be transformed into a grammar G_M such that $L(M) = L(G_M)$.

Small part of English as a context-free language

$$
\begin{aligned}
S = \langle sentence\rangle &\rightarrow \langle noun-phrase\rangle\langle verb-phrase\rangle.\\
\langle sentence\rangle &\rightarrow \langle noun-phrase\rangle\langle verb-phrase\rangle\langle object-phrase\rangle.\\
\langle noun-phrase\rangle &\rightarrow \langle name\rangle \mid \langle article\rangle\langle noun\rangle\\
\langle name\rangle &\rightarrow John \mid Jill\\
\langle noun\rangle &\rightarrow bicycle \mid mango\\
\langle article\rangle &\rightarrow a \mid the\\
\langle verb-phrase\rangle &\rightarrow \langle verb\rangle \mid \langle adverb\rangle\langle verb\rangle\\
\langle verb\rangle &\rightarrow eats \mid rides\\
\langle adverb\rangle &\rightarrow slowly \mid frequently\\
\langle adjective-list\rangle &\rightarrow \langle adjective\rangle\langle adjective-list\rangle \mid \epsilon\\
\langle adjective\rangle &\rightarrow big \mid juicy \mid yellow\\
\langle object-phrase\rangle &\rightarrow \langle adjective-list\rangle\langle name\rangle\\
\langle object-phrase\rangle &\rightarrow \langle article\rangle\langle adjective-list\rangle\langle noun\rangle
\end{aligned}
$$

Jill frequently eats a juicy yellow mango belongs to this language

$G = \langle V, \Sigma, R, S \rangle$ with $S \in V$ and R a set of rules of the form

$$\boxed{A \to wB \text{ or } A \to \lambda}$$

where $A, B \in V$ and $w \in \Sigma$ (so w contains no help symbols!) Such grammar is called *regular*

Many books are a bit more liberal

$$\boxed{A \to wB \text{ or } A \to \lambda \text{ or } A \to w}$$

But that is essentially the same

$A \to w$ can be simulated by $A \to wA'$ and $A' \to \lambda$

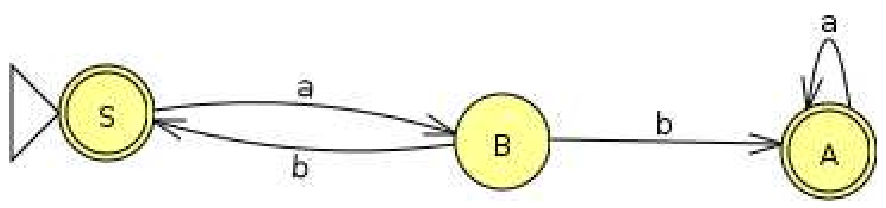
We have already seen

Theorem All regular languages can be generated by regular grammars.

The other way around:
Theorem If G is a regular grammar, then $L(G)$ is a regular language.

Example
$$\begin{array}{lcl} S & \to & \mathtt{a}B \mid \lambda \\ B & \to & \mathtt{b}S \mid \mathtt{b}A \\ A & \to & \mathtt{a}A \mid \lambda \end{array}$$

NFA



The *context sensitive languages* have a *context sensitive grammar* with production rules of the form

$$uXv \to uwv,$$

with $u, v \in (\Sigma \cup V)^*$ arbitrary and $w \in (\Sigma \cup V)^+$ (so $w \neq \lambda$!) or

$$S \to \lambda$$

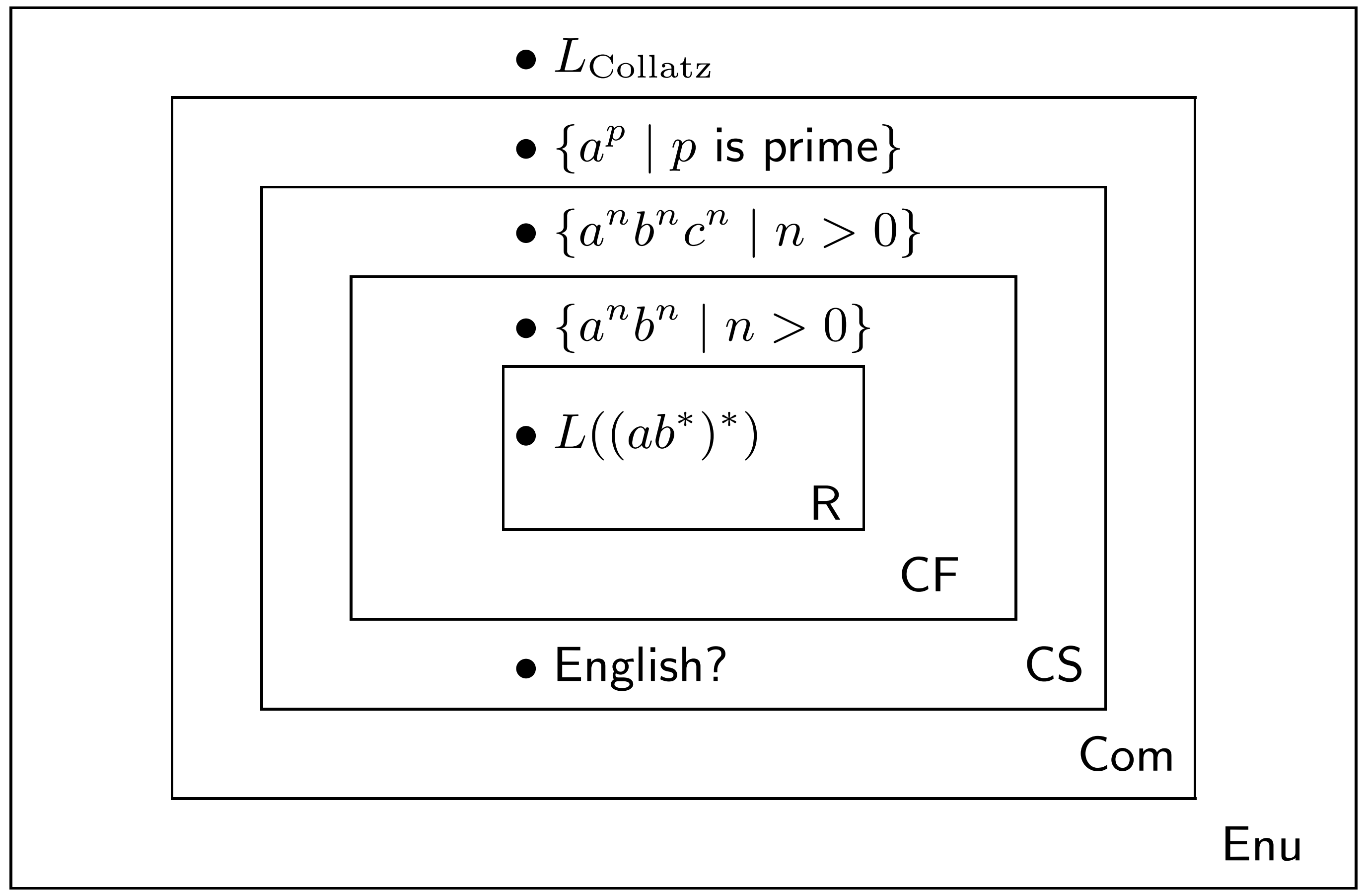
if S doesn't occur on the right of a rule

For the *enumerable languages* the grammar is *unrestricted*, so the production rules are of the form

$$u \to v,$$

with $u, v \in (\Sigma \cup V)^*$.

A languages is called *computable* if both L and $\overline{L} = \Sigma^* - L$ are enumerable. Let R, CF, CS, Com, Enu be notations for the regular, context-free, context-sensitive, computable and enumerable languages, respectively. Then R ⊆ CF ⊆ CS ⊆ Com ⊆ Enu.



The Chomsky hierarchy