BU CS 332 – Theory of Computation

Lecture 7:

- Context-free grammars
- Pumping lemma for CFLs

Reading:

Sipser Ch 2.1, 2.3

Ran Canetti

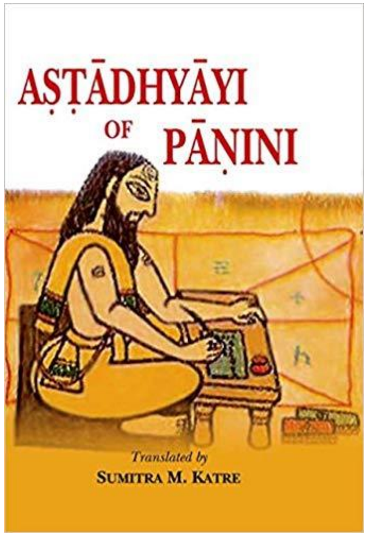
September 24, 2020

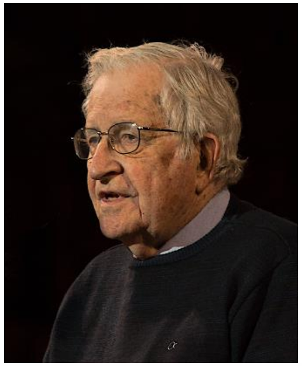
Context-Free Grammars

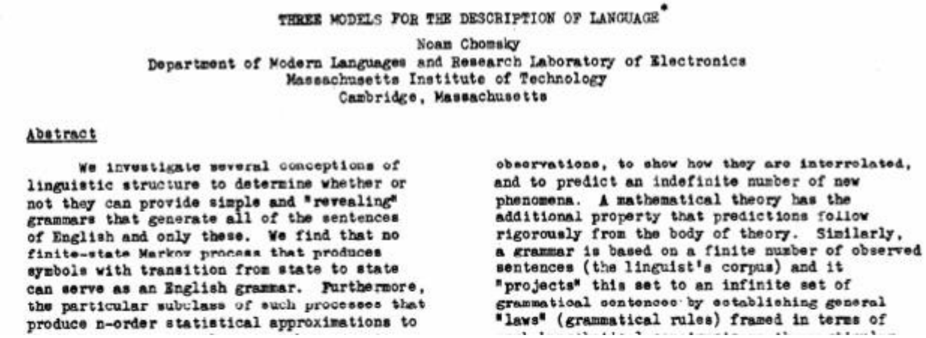
Some History

An abstract model for two distinct problems

Rules for parsing natural languages







THREE MODELS FOR THE DESCRIPTION OF LANGUAGE*

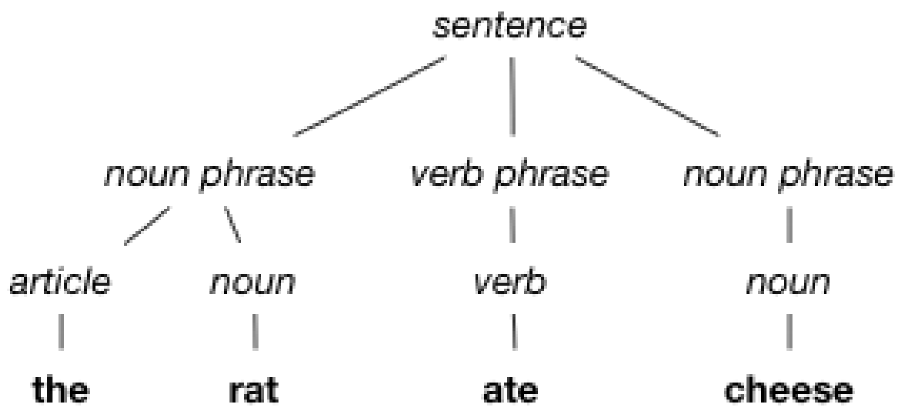
Noam Chomsky
Department of Modern Languages and Research Laboratory of Electronics
Massachusetts Institute of Technology
Cambridge, Massachusetts

Abstract

We investigate several conceptions of linguistic structure to determine whether or not they can provide simple and "revealing" grammars that generate all of the sentences of English and only these. We find that no finite-state Markov process that produces symbols with transition from state to state can serve as an English grammar. Furthermore, the particular subclass of such processes that produce n-order statistical approximations to

observations, to show how they are interrelated, and to predict an indefinite number of new phenomena. A mathematical theory has the additional property that predictions follow rigorously from the body of theory. Similarly, a grammar is based on a finite number of observed sentences (the linguist's corpus) and it "projects" this set to an infinite set of grammatical sentences by establishing general "laws" (grammatical rules) framed in terms of

Parsing an English sentence



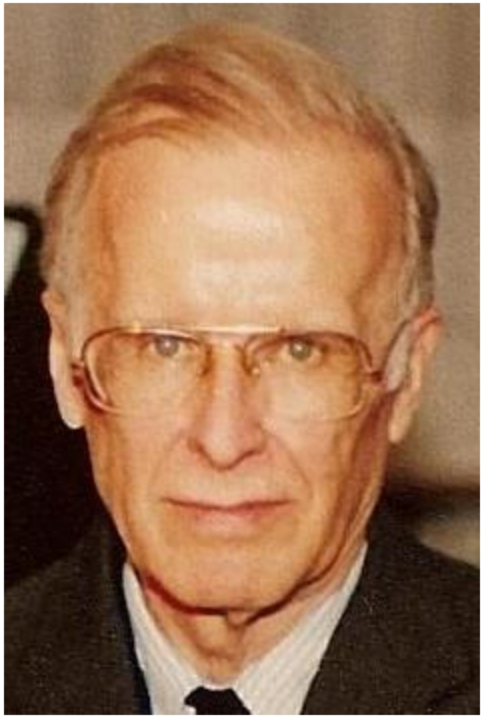
Some History

An abstract model for two distinct problems

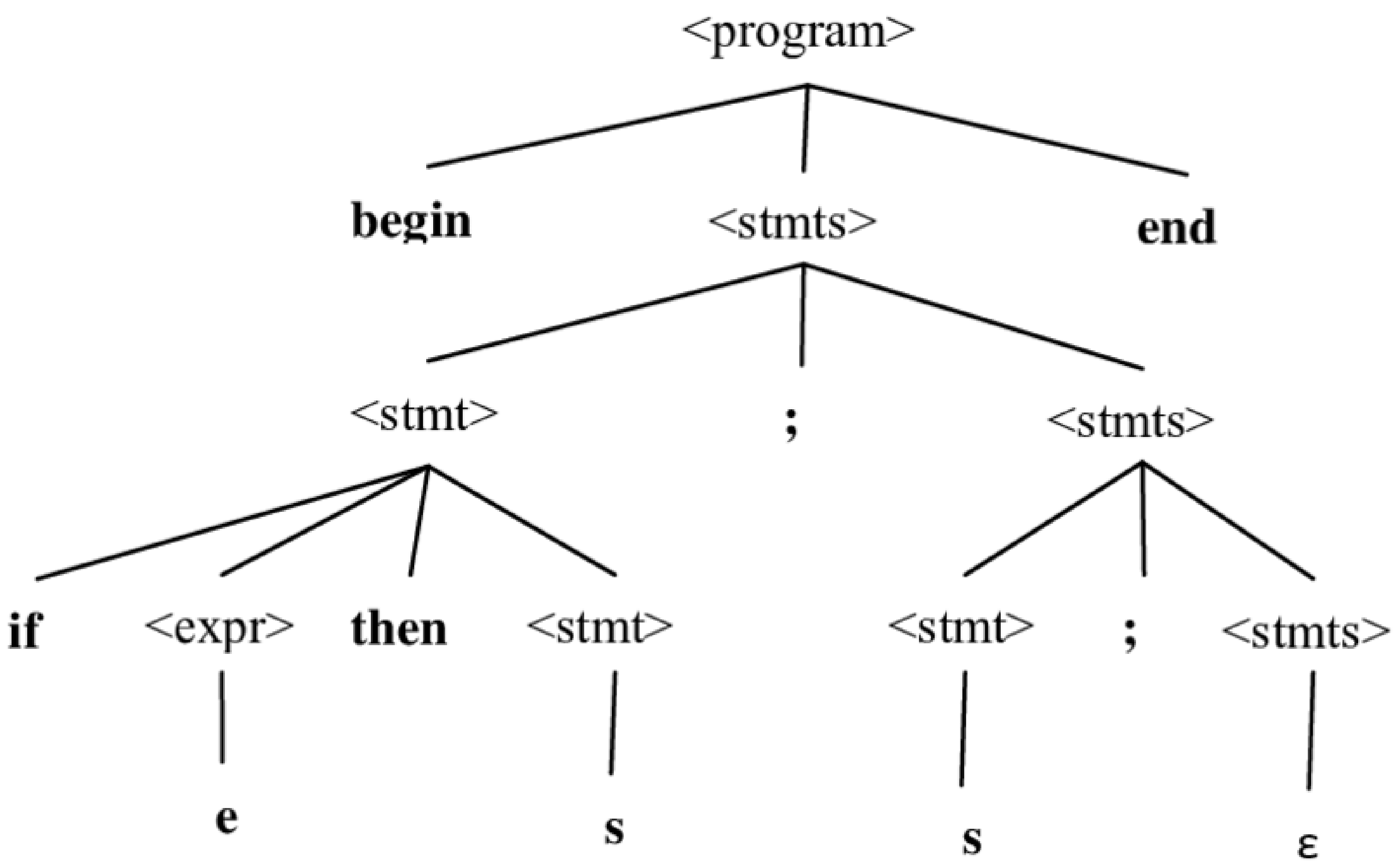
Specification of syntax and compilation for programming languages

1977 ACM Turing Award citation
(John Backus)

For profound, influential, and lasting contributions to the design of practical high-level programming systems, notably through his work on FORTRAN, and for seminal publication of formal procedures for the specification of programming languages.



Parsing a computer program



Context-Free Grammar (Informal)

Example Grammar G

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Derivation

$L(G) =$

Context-Free Grammar (Informal)

Example Grammar G

$$
\begin{aligned}
E &\rightarrow E + T \\
E &\rightarrow T \\
T &\rightarrow T \times F \\
T &\rightarrow F \\
F &\rightarrow (E) \\
F &\rightarrow a \\
F &\rightarrow b
\end{aligned}
$$

Derivation

$L(G) \ =$

Socially Awkward Professor Grammar

<PHRASE> → <FILLER><PHRASE>

<PHRASE> → <START><END>

<FILLER> → LIKE

<FILLER> → UMM

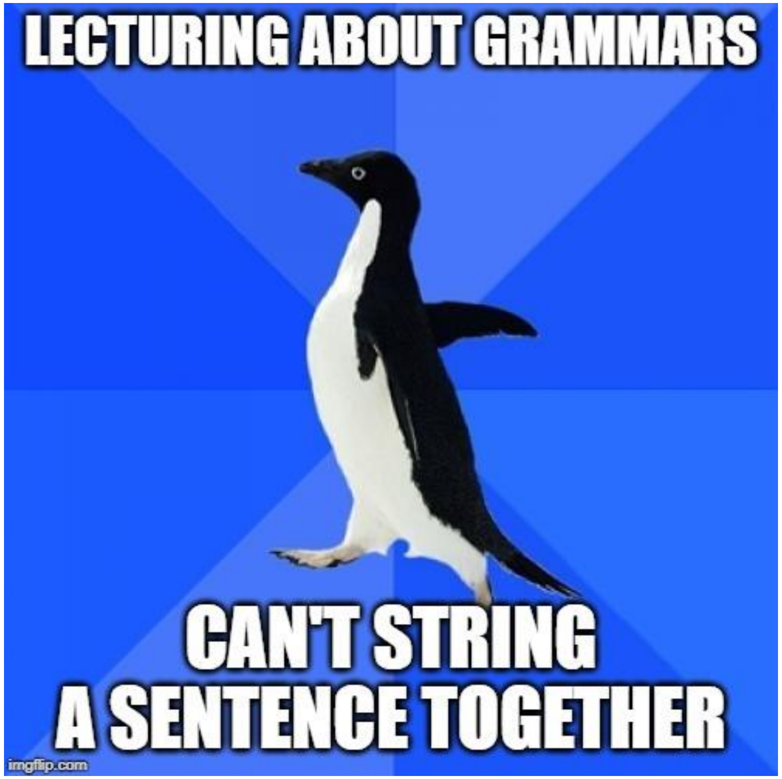
<START> → YOU KNOW

<START> → ε

<END> → WHOOPS

<END> → SORRY

<END> → $#@!



Socially Awkward Professor Grammar

<PHRASE> → <FILLER><PHRASE> | <START><END>

<FILLER> → LIKE | UMM

<START> → YOU KNOW | ε

<END> → WHOOPS | SORRY | $#@!

Socially Awkward Professor Grammar

<PHRASE> → <FILLER><PHRASE> | <START><END>

<FILLER> → LIKE | UMM

<START> → YOU KNOW | ε

<END> → WHOOPS | SORRY | $#@!

Is " YOU KNOW LIKE WHOOPS SORRY" In the language of this grammar?

Context-Free Grammar (Formal)

A CFG is a 4-tuple $G = (V, \Sigma, R, S)$

- V is a finite set of variables
- Σ is a finite set of terminal symbols (disjoint from V)
- R is a finite set of production rules of the form $A \to w$, where $A \in V$ and $w \in (V \cup \Sigma)^*$
- $S \in V$ is the start variable

Example: $G = (\{S\}, \Sigma, R, S)$

where

$$\Sigma = \{a, b\}$$
$$R = \{S \to aSb, S \to \varepsilon\}$$

Context-Free Grammar (Formal)

A CFG is a 4-tuple $G = (V, \Sigma, R, S)$

V = variables Σ = terminals R = rules S = start

- A *state* is a sequence of variables and terminals

Context-Free Grammar (Formal)

A CFG is a 4-tuple $G = (V, \Sigma, R, S)$

V = variables Σ = terminals R = rules S = start

- A *state* is a sequence of variables and terminals
- We say that state a derives state b ($a \Rightarrow b$) if n is obtained be applying a rule to one of the variables in a.

  eg, $uAv \Rightarrow uwv$ ("uAv yields uwv") if $A \rightarrow w$ is a rule of the grammar.

Context-Free Grammar (Formal)

A CFG is a 4-tuple $G = (V, \Sigma, R, S)$

V = variables Σ = terminals R = rules S = start

- A *state* is a sequence of variables and terminals
- We say that state a derives state b ($a \Rightarrow b$) if n is obtained be applying a rule to one of the variables in a.

  eg, $uAv \Rightarrow uwv$ ("uAv yields uwv") if $A \rightarrow w$ is a rule of the grammar.

- We say a $\overset{*}{\Rightarrow} b$ ("a derives b") if a$=$ b or there exists a sequence such that a$\Rightarrow$ $a_1 \Rightarrow a_2 \Rightarrow \cdots \Rightarrow b$

Context-Free Grammar (Formal)

A CFG is a 4-tuple $G = (V, \Sigma, R, S)$

V = variables Σ = terminals R = rules S = start

- A *state* is a sequence of variables and terminals
- We say that state a derives state b ($a \Rightarrow b$) if n is obtained be applying a rule to one of the variables in a.

  eg, $uAv \Rightarrow uwv$ (“uAv yields uwv”) if $A \rightarrow w$ is a rule of the grammar.

- We say a $\overset{*}{\Rightarrow} b$ (“a derives b”) if a$=$ b or there exists a sequence such that a$\Rightarrow$ $a_1 \Rightarrow a_2 \Rightarrow \cdots \Rightarrow b$

- Language of the grammar: $L(G) = \{w \in \Sigma^* | S \overset{*}{\Rightarrow} w\}$

Example: $G = (\{S\}, \Sigma, R, S)$ where $R = \{S \rightarrow uSv, S \rightarrow \varepsilon\}$

$$L(G) = \{u^n v^n | n \geq 0\}$$

CFG Examples

Give context-free grammars for the following languages

1. The empty language

2. Strings of properly nested parentheses

3. Strings with equal # of a's and b's

Context-Free Languages

L is a *context-free language* if it is the language of some CFG

Questions about CFLs

1. Which languages are *not* context-free?
2. How do we recognize whether $w \in L$?
3. What are the closure properties of CFLs?

Pumping Lemma for context-free languages

Let L be a context-free language.

Then there exists a "pumping length" p such that

For every $w \in L$ where $|w| \geq p$,

w can be split into five parts $w = uvxyz$ where:

1. $|vy| > 0$
2. $|vxy| \leq p$
3. $uv^ixy^iz \in L$ for all $i \geq 0$

Pumping Lemma example

Claim: $L = \{a^n b^n c^n | n \geq 0\}$ is not context-free

Proof: Assume L is context-free with pumping length p

1. Find $w \in L$ with $|w| \geq p$
2. Show that w cannot be pumped
   If $w = uvxyz$ with $|vy| > 0, |vxy| \leq p$, then...

Case 1: v, y both contain only one kind of symbol

Case 2: Either v or y contains two kinds of symbols

Pumping Lemma example

Claim: $L = \{a^n b^n c^n | n \geq 0\}$ is not context-free

Proof: Assume L is context-free with pumping length p

1. Find $w \in L$ with $|w| \geq p$
2. Show that w cannot be pumped

   If $w = uvxyz$ with $|vy| > 0, |vxy| \leq p$, then…

Case 1: v, y both contain only one kind of symbol

Pumping Lemma example

Claim: $L = \{a^n b^n c^n | n \geq 0\}$ is not context-free

Proof: Assume L is context-free with pumping length p

1. Find $w \in L$ with $|w| \geq p$
2. Show that w cannot be pumped
   If $w = uvxyz$ with $|vy| > 0, |vxy| \leq p$, then…

Case 2: Either v or y contains two kinds of symbols

Pumping Lemma: Proof idea

Let L be a context-free language. If $w \in L$ is long enough, then every parse tree for w has a repeated variable.

Pumping Lemma Proof

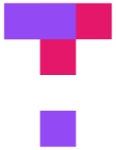
What does "long enough" mean? (How do we choose the pumping length p?)

- Let G be a CFG for L
- Suppose the right-hand side of every rule in G uses at most b symbols
- Let $p = b^{|V|+1}$

Claim: If $w \in L$ with $|w| \geq p$, then the smallest parse tree for w has height at least $|V| + 1$

Pumping Lemma Proof

Claim: If $w \in L$ with $|w| \geq p$, then the smallest parse tree for w has height at least $|V| + 1$



- By the pigeonhole principle, there is a path down the parse tree with a repeated variable R
- Choose two such occurrences within the bottom $|V| + 1$ levels

Context-Free Languages

L is a *context-free language* if it is the language of some CFG

Questions about CFLs

1. Which languages are *not* context-free?
2. How do we recognize whether $w \in L$?
3. What are the closure properties of CFLs?

Pumping Lemma

Pumping Lemma for regular languages

Let L be a regular language.

Then there exists a "pumping length" p such that

For every $w \in L$ where $|w| \geq p$,

w can be split into three parts $w = xyz$ where:

1. $|y| > 0$
2. $|xy| \leq p$
3. $xy^iz \in L$ for all $i \geq 0$

Pumping Lemma for context-free languages

Let L be a context-free language.

Then there exists a "pumping length" p such that

For every $w \in L$ where $|w| \geq p$,

w can be split into five parts $w = uvxyz$ where:

1. $|vy| > 0$
2. $|vxy| \leq p$
3. $uv^i xy^i z \in L$ for all $i \geq 0$

Example:

$L = \{w \in \{0,1\}^* | w = w^R\}$

$w = 0$

Pumping Lemma for context-free languages

Let L be a context-free language.

Then there exists a “pumping length” p such that

For every $w \in L$ where $|w| \geq p$,

w can be split into five parts $w = uvxyz$ where:

1. $|vy| > 0$
2. $|vxy| \leq p$
3. $uv^i xy^i z \in L$ for all $i \geq 0$

Example:

$L = \{w \in \{0,1\}^* | w = w^R\}$

$w = 010$

Pumping Lemma as a game

1. YOU pick the language L to be proved non context-free.
2. ADVERSARY picks a possible pumping length p.
3. YOU pick w of length at least p.
4. ADVERSARY divides w into u, v, x, y, z, obeying rules of the Pumping Lemma: $|vy| > 0$ and $|vxy| \leq p$.
5. YOU win by finding $i \geq 0$, for which uv^ixy^iz is not in L.

If *regardless* of how the ADVERSARY plays this game, you can always win, then L is non context-free

Pumping Lemma example

Claim: $L = \{a^n b^n c^n | n \geq 0\}$ is not regular

Proof: Assume L is regular with pumping length p

1. Find $w \in L$ with $|w| \geq p$
2. Show that w cannot be pumped

   If $w = uvxyz$ with $|vy| > 0, |vxy| \leq p$, then...

Pumping Lemma example

Claim: $L = \{a^n b^n c^n | n \geq 0\}$ is not regular

Proof: Assume L is regular with pumping length p

1. Find $w \in L$ with $|w| \geq p$
2. Show that w cannot be pumped

   If w = $uvxyz$ with $|vy| > 0, |vxy| \leq p$, then...

Pumping Lemma example

<u>Claim:</u> $L = \{a^n b^n c^n | n \geq 0\}$ is not regular

<u>Proof:</u> Assume L is regular with pumping length p

1. Find $w \in L$ with $|w| \geq p$
2. Show that w cannot be pumped

   If w = $uvxyz$ with $|vy| > 0, |vxy| \leq p$, then...