Generative Grammars

Generative Grammars Informally

- Fix a vocabulary
  - A set of symbols
  - Some of these symbols, called **terminals**, play the tokens of the output stream of lexical analysis

- E.g. take the vocabulary {S, a, b} where “a” and “b” are terminals

Generative Grammars

- One non-terminal symbol of the vocabulary is chosen as **start symbol**

- E.g., S in {S, a, b}

Generative Grammars

- Fix a set of **productions**
  - Rules for rewriting strings into strings
  - Constraint:
    - the string to be replaced must contain at least a non-terminal

- E.g. $\{S \rightarrow aSb, S \rightarrow ab\}$

Generative Grammars

- These are the ingredients of a **generative grammar**
- A **language** of words of terminals can be generated from the start symbol:
  - Apply the rewriting rules in any possible way, as many times as possible
  - Each rewriting is called a **derivation step**

Notation for the derivation relation

→ aSb, S → ab}

S ⇒ a b

- Is a one-step derivation from S
- “ab” is made up of terminals only
- Hence “ab” **belongs** to the language generated by the given grammar

{S → aSb, S → ab}

$S \Rightarrow aSb \Rightarrow aabb$

- Is a two-step derivation from S
- “aabb” is made up of terminals only
- Hence “aabb” **belongs** to the language generated by the given grammar

$\{S \rightarrow aSb, S \rightarrow ab\}$

$S \Rightarrow aSb \Rightarrow aaSbb$

- Is a two-step derivation of a string from S
- But “aaSbb” contains a non-terminal
- Hence “aaSbb” **does not belong** to the language generated by the given grammar

$\{S \to aSb, S \to ab\}$

- Which is the language generated by this grammar?

- $\{a^n b^n \mid n>0\}$

Notation

- Capital letters for non-terminals

Convention

- Special character epsilon (ε) used to denote the empty word

- Length of ε is 0
  - ε ε ε
  - ε b^0 for every terminal b

Example 1

$S \to aAb$

$aA \to aaAb$

$A \to \varepsilon$

- Generated language: $\{a^n b^n \mid n>0\}$
- OBSERVE: Different grammars can generate the same language

Example 2

$S \rightarrow AB$

$A \rightarrow aA$

$A \rightarrow a$

$B \rightarrow Bb$

$B \rightarrow b$

- Generated language: $\{a^n b^m \mid n,m>0\}$

Example 3

$S \rightarrow aSBc$

$S \rightarrow abc$

$cB \rightarrow Bc$

$bB \rightarrow bb$

- Generated language: $\{a^n b^n c^n \mid n>0\}$

Example 4

$S \rightarrow AB$

$A \rightarrow a$

- Generated language:

Example 5

S →

– Generated language:

Example 6

$S \rightarrow aSb$

$S \rightarrow \varepsilon$

- Generated language:
  - $\{a^n b^n \mid n>0\} = \{a^n b^n \mid n0\}$

Examp

Notation for more productions with same left-hand side

$S \rightarrow CD$

$C \rightarrow aCA \mid bCB$

$AD \rightarrow aD$

$BD \rightarrow bD$

$Aa \rightarrow aA$

$Ab \rightarrow bA$

$Ba \rightarrow aB$

$Bb \rightarrow bB$

$C \rightarrow \varepsilon$

$D \rightarrow \varepsilon$

– Generated language?

Example 7: Derivation 1

$S \rightarrow CD$

$C \rightarrow aCA \mid bCB$

$AD \rightarrow aD$

$BD \rightarrow bD$

$Aa \rightarrow aA$

$Ab \rightarrow bA$

$Ba \rightarrow aB$

$Bb \rightarrow bB$

$C \rightarrow \varepsilon$

$D \rightarrow \varepsilon$

$S \Rightarrow CD$

$CD \Rightarrow D$

$D \Rightarrow \varepsilon$

Example 7: Derivation 2

$S \rightarrow CD$

$C \rightarrow aCA \mid bCB$

$AD \rightarrow aD$

$BD \rightarrow bD$

$Aa \rightarrow aA$

$Ab \rightarrow bA$

$Ba \rightarrow aB$

$Bb \rightarrow bB$

$C \rightarrow \varepsilon$

$D \rightarrow \varepsilon$

$S \Rightarrow CD$

$CD \Rightarrow aCAD$

$aCAD \Rightarrow aCaD$

$aCaD \Rightarrow aaD$

$aaD \Rightarrow aa$

Example 7: Derivation 3

$S \to CD$

$C \to aCA \mid bCB$

$AD \to aD$

$BD \to bD$

$Aa \to aA$

$Ab \to bA$

$Ba \to aB$

$Bb \to bB$

$C \to \varepsilon$

$D \to \varepsilon$

$S \Rightarrow CD$

$CD \Rightarrow aCAD$

$aCAD \Rightarrow abCBAD$

$abCBAD \Rightarrow abCBA$

$abCBA \Rightarrow abBA$

Example 7: Derivation 4

$S \to CD$

$C \to aCA \mid bCB$

$AD \to aD$

$BD \to bD$

$Aa \to aA$

$Ab \to bA$

$Ba \to aB$

$Bb \to bB$

$C \to \varepsilon$

$D \to \varepsilon$

$S \Rightarrow CD$

$CD \Rightarrow aCAD$

$aCAD \Rightarrow abCBAD$

$abCBAD \Rightarrow abCBaD$

$abCBaD \Rightarrow abCaBD$

$abCaBD \Rightarrow abCabD$

$abCabD \Rightarrow ababD$

$ababD \Rightarrow abab$

Generative Grammars Formally

- A grammar is a tuple

  (V,T,S,P)

  - V vocabulary of terminals and non-terminals
  - T set of terminals
  - S start symbol in (V\T)
  - P set of productions

Not

Zero or more repetitions of elements in the base set

- Uppercase, early in the abet
  - A,B,.... (V \ T)
- Uppercase, late in tl lphabet
  - X,Y,... V
- Lowercase, early the alphabet
  - a,b,.... T
- Lowercase, early in Greek alphabet
  - V*
- Strings of terminals
  - w, w_0,....

Productions

- General form:

One or more repetitions of elements in the base

- V^+
- contains at least a non-terminal
- called **driver** of the production
- called **body** of the production

Generated Languages

- $G = (V,T,S,P)$

- $\mathbf{L(G) = \{\, w \mid w\ T^* \text{ and } S \Rightarrow^* w \,\}}$

T* because w may just be

Hierarchy of Grammars

- Depending on the shape of productions

- **Context-free grammars**, or just **free grammars**:

Context-free Languages

- L is a **context-free language**
- Iff
- There exists a context-free grammar G such that L=L(G)

Context-free Languages

Our focus

Canonical Derivations

- **Rightmost** (**Leftmost**) derivation step:
  - Replace the rightmost (**leftmost**) non-terminal
- **Canonical derivations** of words in the language:
  - Either every step is rightmost
  - Or every step is leftmost

Derivation Trees

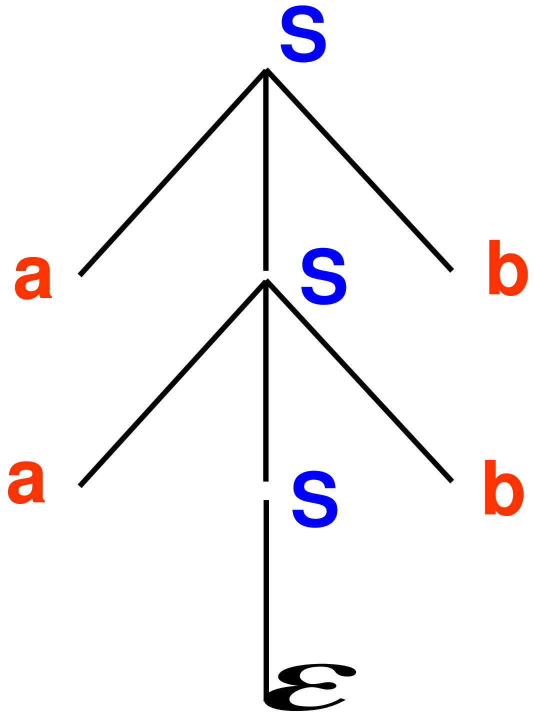
- Start symbol is the root
- For every derivation step under the production
- A X_1 X_2 … X_n
- Generate children X_1 X_2 … X_n for node A
- Terminals are the leaves (and so is)

Derivation Trees

- The derived word is at the **frontier** of the tree

Example

S → aSb | ε



Ambiguity in Natural Languages

- L’uomo guarda la donna con il binocolo

Ambiguity

- Grammar G is **ambiguous**
- Iff
- There exists w L(G) that can be generated by two distinct canonical derivations, either both rightmost or both leftmost

Arithmetic Expressions

$E \to E{+}E \mid E{*}E \mid n$

- Ambiguous?

Arithmetic Expressions

$E \rightarrow E+E \mid E*E \mid n$

- Take $w = n+n*n$

Arithmetic Expressions

E

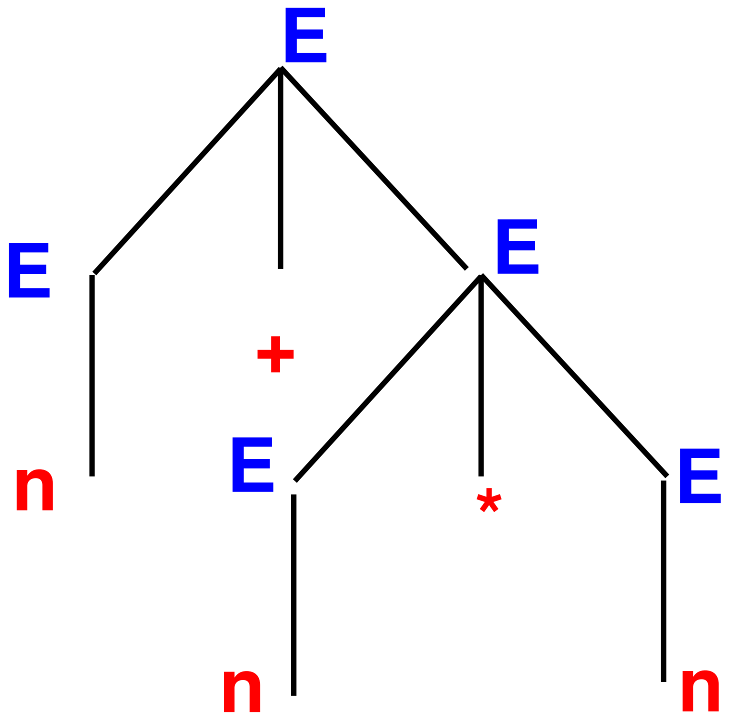
E + E

n + E

n + E * E

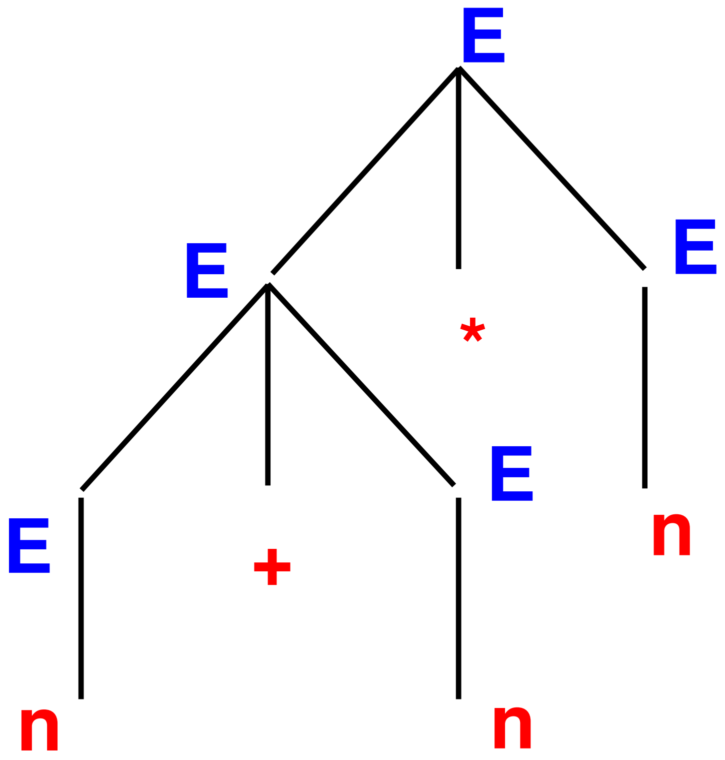
n + n * E

n + n * n

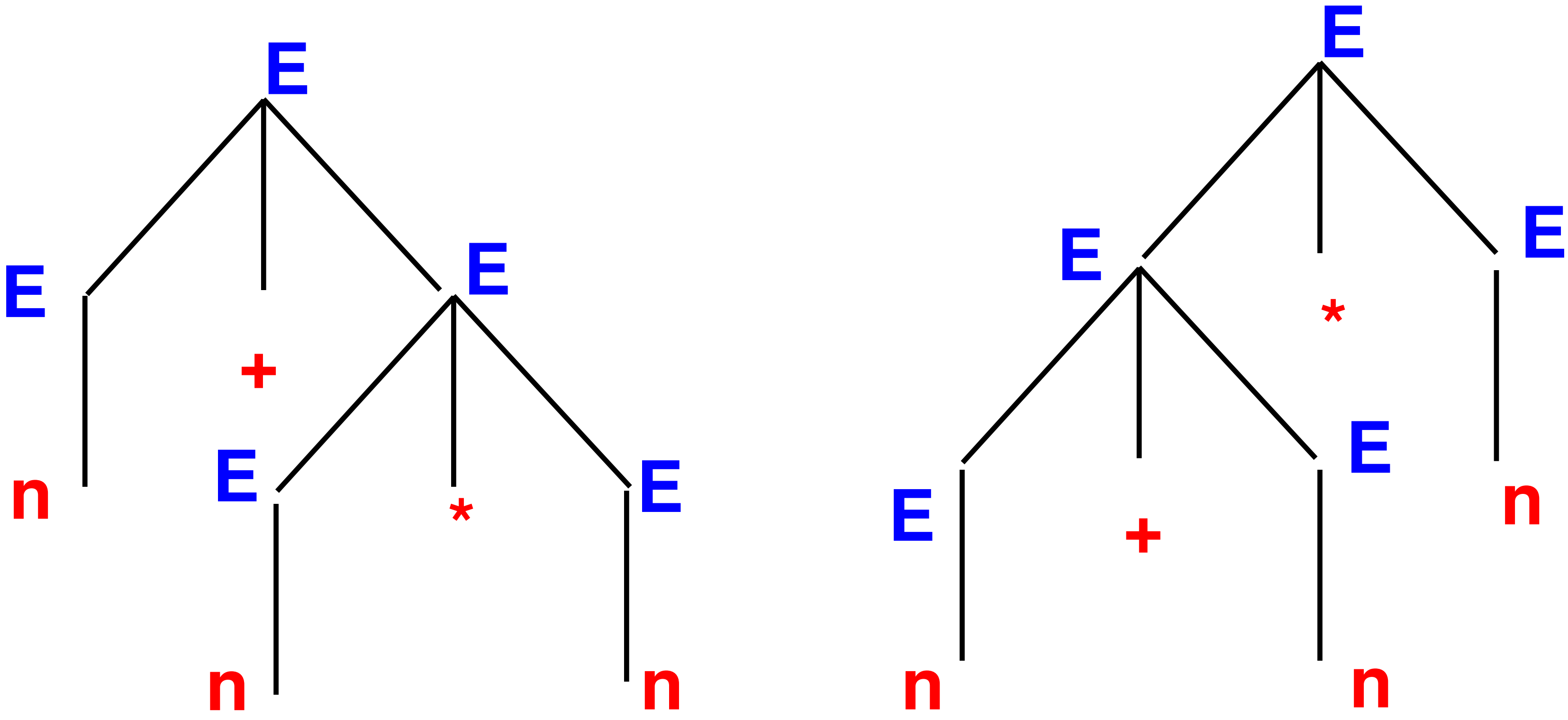


Arithmetic Expressions

But also



Arithmetic Expressions



Arithmetic Expressions

$E \rightarrow E+E \mid E*E \mid n$

– Ambiguous!

Dangling Else

$S \to$ if b then S | if b then S else S | other

- Ambiguous?

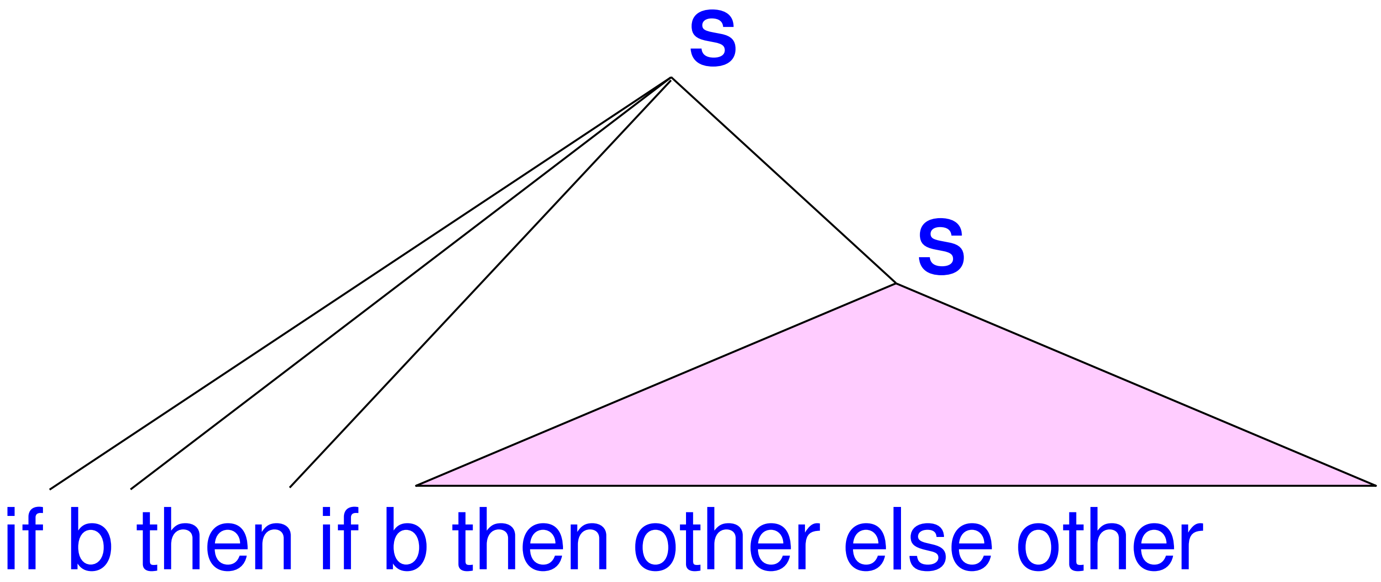
Dangling Else

S → if b then S | if b then S else S | other

- Take
- w = if b then if b then other else other

- Which “then” matches “else”?

Dangling Else

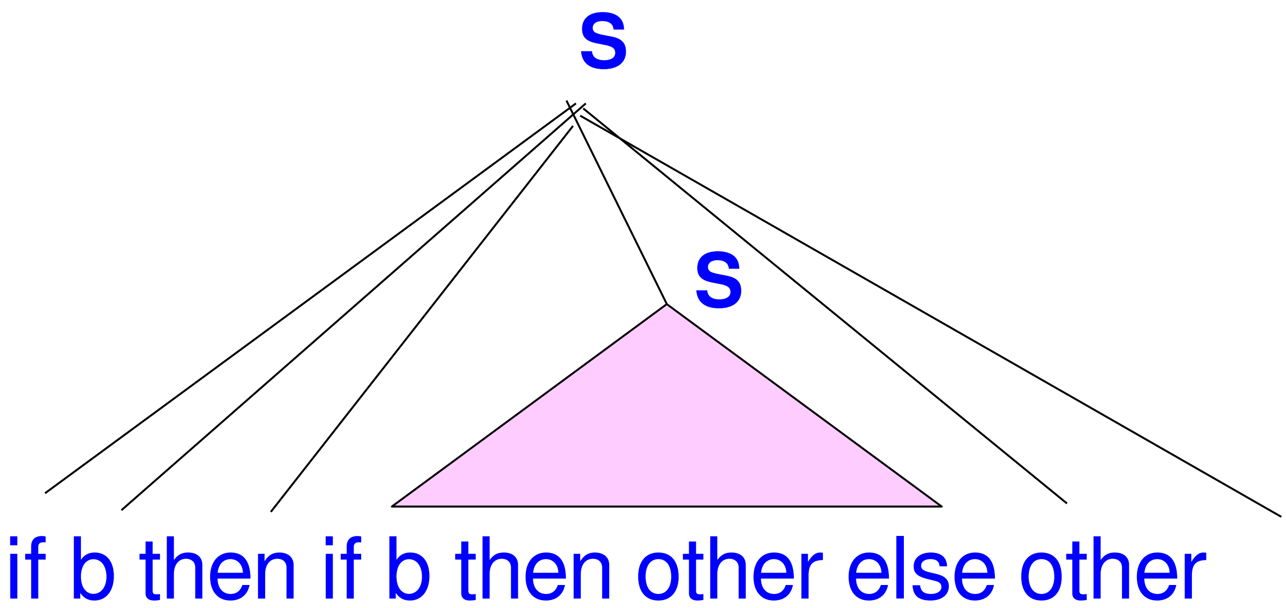


s

s

if b then if b then other else other

Dangling Else



Dangling Else

S → if b then S | if b then S else S | other

– Ambiguous!

Observation

- Ambiguity is undecidable

- No algorithm can be designed to decide whether a grammar is ambiguous or not