Introduction to grammars: outline

- Introduction to grammars
- Phrase Structure Grammar and language
  - Derivation

Introduction to grammars

- A formal language is a collection of strings over ∑ with some rules known as grammars.
- Grammar rules can be represented using a syntax diagram.
- Alternatively BNF (Backus-Naur Form) notation can be used.
- BNF uses the following:
  - Nonterminals represented enclosed by <>. Terminals are represented as they are.
  - { } - represent repetition of nonterminals, terminals zero or more times
  - ::= stands for “is defined as”
  - | stands for OR
  - () used to group symbols

Ex. <identifier> ::= <letter>|<letter>{<letter>|<digit>}

<letter> ::= a|b|c|…

<digit> ::= 0|1|2|…|9

Introduction to grammars: cont'd

- Phrase Structure Grammar (PSG)
  - Definition: A PSG is a 4-tuple (N, T, P, S) where:
    a. N is a finite set of nonterminals
    b. T is a final set of terminals
    c. P is a finite set of productions /rules/ of the form $\alpha \rightarrow \beta$, where α and β are strings on N U T and α should contain at least one symbol from N.
    d. S Є N is the start symbol of the grammar.

Note: The right hand side production, β, can be an empty string. Such a production is called a λ-production.

Ex. G = (N, T, P, S) = ({S, B, C}, {a, b, c}, P, S)

where P is given by:

$$S \rightarrow aSBC \mid aBC$$
$$BC \rightarrow CB$$
$$aB \rightarrow ab$$
$$C \rightarrow Cc \mid \lambda$$

Introduction to grammars: cont'd

- Derivation
  1. If α generates β, then we write $\alpha \Rightarrow \beta$
  2. If $\alpha_1 \Rightarrow \alpha_2$, $\alpha_2 \Rightarrow \alpha_3$, …, $\alpha_{n-1} \Rightarrow \alpha_n$, then we write $\alpha_1 \Rightarrow \alpha_2 \Rightarrow \alpha_3 \Rightarrow \ldots \Rightarrow \alpha_n$ or $\alpha_1 \overset{+}{\Rightarrow} \alpha_n$
- Let G = (N, T, P, S) be a grammar, if S => α in zero or more steps, $\alpha \in (N \cup T)^*$, then α is called a **sentential form**.
- A **sentence** (in G) is a sentential form in T*.
- The language generated from the grammar G is denoted by L(G). $L(G) = \{x \in T^* \mid S \overset{*}{\Rightarrow} x\}$

  i.e. L(G) is the set of all terminal strings derived from the start symbol S.

Introduction to grammars: cont'd

- Ex1.
  G = (N, T, P, S) where:
  N = {<sentence>, <noun>, <verb>, <adverb>}
  T = {Sam, Dan, ate, sang, well}
  S = <sentence>
  P consists of:
  <sentence> → <noun><verb> | <noun><verb><adverb>
  <noun> → Sam | Dan
  <verb> → ate | sang
  <adverb> → well

Introduction to grammars: cont'd

- Ex2.
  S → A|B
  A → aA|bB|a|b
  B → bB|b

- Ex3.
  S → a|bS

- Ex4.
  S → aA|bB|a|b
  A → aA|a
  B → bB|b

Introduction to grammars: cont'd

- Note: that reverse derivation is not permitted. For instance, if S $\rightarrow$ AB is a production, then we can replace S by AB, but we cannot replace AB by S.
- Notations:
  1. If A is any set, then A* denotes the set of all strings over A and $A^+ = A^* - \{\lambda\}$
  2. A, B, C, A1, A2, … denote nonterminals
  3. a, b, c, … denote terminals
  4. x, y, z, w, … denote strings of terminals
  5. α, β, … denote strings from (N U T)*
  6. If A $\rightarrow$ α is a production where A Є N, the production is called an A-production
  7. If $A \rightarrow \alpha_1$, $A \rightarrow \alpha_2$, $A \rightarrow \alpha_3$, $A \rightarrow \alpha_4$ … $A \rightarrow \alpha_n$ are all A-productions, these can be written as $A \rightarrow \alpha_1 | \alpha_2 | \alpha_3 | \alpha_4 | \ldots \alpha_n$
  8. $X^0 = \lambda$ for any symbol X Є N U T

Introduction to grammars: cont'd

- <u>Definition</u>: Let G1 and G2 be two grammars, then G1 and G2 are **equivalent** if L(G1) = L(G2).
- Ex5. G = ({S}, {a}, {S → SS | a}, S). Find L(G)
- Ex6. G = ({S, C}, {a, b}, P, S) where P is given by:

  S → aCa

  C → aCa | b

  Find L(G)
- Ex7. G = ({S}, {a}, {S → aS|a}, S). Find L(G)
- Ex8. Let L be the set of all palindromes over {a, b, 1}. Construct a grammar G that generates L.

  Hint: Use the following recursive definition

  i. λ is a palindrome
  ii. a, b are palindromes
  iii. If x is a palindrome, axa and bxb are palindromes