Lesson #2: Context Free Grammars

What's It All About?

There are several types of "phrase structure grammars", grammars grounded in the concept of repeatedly replacing one string of symbols by another string of symbols. The "context free grammar" is just one of them. That said, it is a very popular type of phrase structure grammar, due to its simplicity, and, particularly with suitable augmentations, its broad applicability.

This lesson presents an introduction to context free grammars.

Context Free Grammar: The definition

A **context free grammar (CFG)** is defined in terms of four constituent parts:

1. A set of **terminal symbols**, which are symbols that appear in the language being defined.
2. A set of **nonterminal symbols**, which do not appear in the language being defined, but help to define the language of interest by denoting particular strings of symbols.
3. A set of **productions**, or **rewrite rules**, of the form:
   NONTERMINAL → LIST-OF-SYMBOLS
4. A **start symbol**, which is a nonterminal symbol that, in an abstract sense, represents the language being defined.

Example: CFG for the Ladida language

The following CFG is intended to define the Ladida language:

1. Terminals = {LA, DI, DA}
2. Nonterminals = {SENTENCE, M, X}
3. Productions =

   (1) SENTENCE → LA DA

   (2) SENTENCE → LA M DA

   (3) M → X

   (4) M → M X

   (5) X → LA

   (6) X → DI

   (7) X → DA

4. Start symbol: SENTENCE

Thoughts on CFGs

1. CFGs can be thought of as a knowledge representation with a declaritive part and and a procedural part. So far, we have just focussed on the declarative part.
2. The procedural part? That pertains to using the declarative part either to **recognize** if a form is a sentence in the language being defined, or to **generate** sentences in the language being defined.
3. Clearly, a CFG is a **model** of a language!

The language generated by a CFG

CFGs are a kind of **generative grammar**, in that they can be used to generate all of the sentences in the language that they define. This generative aspect of CFGs is at the heart of the procedural part of CFGs alluded to in the previous item.

Just how one makes use of the four declarative elements of a CFG in order to produce the sentences of the language generated by the grammar is specified by the following definition:

Given a context free grammar G, the language generated by a G, L(G), is the set of all strings of terminal symbols that are derivable from the start symbol.

Refining the definition

What does it mean, "derivable"? String x is **derivable** from string s if there is a sequence of direct derivations beginning with s and ending with x.

What does it mean, "direct derivation"? Given (1) a string of symbols SS which contains nonterminal symbol LHS, and (2) a production with left hand side LHS and right hand side RHS, then the symbol string ZZ which is obtained by substituting RHS for LHS in SS is a **direct derivation** of the symbol string SS by means of the production LHS → RHS.

Example direct derivations

Consider the Ladida CFG. Each of the following is a direct derivation in the context of the grammar:

1. "L M DA" is a **direct derivation** from symbol string "SENTENCE" by means of production "SENTENCE → LA M DA". Alternatively, we can say that "SENTENCE" **directly derives** "L M DA" by means of this production. This is conventionally written as: SENTENCE ⇒ LA M DA

2. “LA X DA” is a **direct derivation** from symbol string “LA M DA” by means of production “M → X”. Alternatively, we can say that “LA M DA” **directly derives** “LA X DA” by means of this production. This is conventionally written as: LA M DA ⇒ LA X DA
3. “LA M X DA” is a **direct derivation** from symbol string “LA M DA” by means of production “M → M X”. Alternatively, we can say that “LA M DA” **directly derives** “LA M X DA” by means of this production. This is conventionally written as: LA M DA ⇒ LA M X DA
4. “LA X X LA X X DA” is a **direct derivation** from symbol string “LA X X X X X DA” by means of production “X → LA”. Alternatively, we can say that “LA X X X X X DA” **directly derives** “LA X X LA X X DA” by means of this production. This is conventionally written as: LA X X X X X DA ⇒ LA X X LA X X DA

Example derivations

Again, consider the Ladida CFG. Each of the following is a derivation in the context of the grammar:

1. Since “LA M DA” ⇒ “LA M X DA” ⇒ “LA M M X DA” ⇒ “LA M X X DA” ⇒ “LA X X X DA”, we can say that: “LA X X X DA” is a **derivation** of “LA M DA”, or that “LA M DA” **derives** “LA X X X DA”
2. Since “SENTENCE” ⇒ “LA M DA” ⇒ “LA X DA” ⇒ “LA DI DA”, we can say that: “LA DI DA” is a **derivation** of “SENTENCE”, or that “SENTENCE” **derives** “LA DI DA”

Please note that, according to the definition of language defined by a CFG, the second example demonstrates that “LA DI DA” is a sentence in the langauge defined by the grammar!

Two questions ...

1. What does the symbol → denote?
2. What does the symbol ⇒ denote?

Please take care to use the symbols appropriately!

Conventional formatting of derivations

There is a conventional way to format derivations. This is it:

1. Place the first direct derivation on a line of its own.
2. There after, break the direct derivation up over two lines, in such a manner that all direct derivation symbols are vertically alligned, and “vertically read”.

Also, it is generally considered good form to reference the production used on each line in a parenthetical elaboration.

Two example derivations ...

The following two examples show that "LA DI DA" and "LA DI DA DI DA" are sentences in the Ladida language.

Example 1

```
SENTENCE ==> LA M DA (production 2)
         ==> LA X DA (production 3)
         ==> LA DI DA (production 6)
```

Example 2

```
SENTENCE ==> LA M DA (production 2)
         ==> LA M X DA (production 3)
         ==> LA M DI DA (production 6)
         ==> LA M X DI DA (production 3)
         ==> LA M DA DI DA (production 7)
         ==> LA X DA DI DA (production 2)
         ==> LA DI DA DI DA (production 6)
```

Exercise: Two Ladida Tasks

Please do the following two things:

1. By means of a derivation which consistently replaces the **leftmost nonterminal** in each string, show that "LA LA DA DA" is a sentence in the Ladida language.
2. Write down all of the sentences in the Ladida language which are no longer than 4 tokens in length.

Notes on CFGs

1. By convention, the nonterminal on the left hand side of the first production is the start symbol for a context free grammar. You should adhere to the convention!
2. You know that something is a nonterminal if it appears on the left hand side of some production.
3. You know that something is a terminal if it does not appear on the left hand side of any production.
4. CFGs are regularly presented by simply listing the productions, leaving it to the reader to infer the remaining three components of the grammar, the terminal set, the nonterminal set, and the start symbol.

CFG for WFFs

The following context free grammar, presented solely in terms of its productions, defines the language of "well formed formulas", or "WFFs" in the propositional calculus.

1. wff → atom

2. wff → conjunction
3. wff → disjunction
4. wff → implication
5. wff → equivalence
6. wff → negation
7. atom → P
8. atom → Q
9. atom → R
10. atom → S
11. conjunction → (wff ∧ wff)
12. disjunction → (wff ∨ wff)
13. implication → (wff → wff)
14. equivalence → (wff ↔ wff)
15. negation → (∼ wff)

Components of the WFF CFG

With respect to the CFG for WFFs, please write down ...

1. The start symbol.
2. The set of terminal symbols.
3. The set of nonterminal symbols.
4. The number of productions.

Derivation of R

Please show that

```
R
```

is a is a sentence in the language of WFFs by deriving it from the start symbol in the given WFF defining grammar.

Derivation of `( ∼ R )`

Please show that

```
( ∼ R )
```

is a is a sentence in the language of WFFs by deriving it from the start symbol in the given WFF defining grammar.

Exercise: English Like Language - ELL

Please consider the following **partial** context free grammar. As you do, note that: (1) the symbol ϵ is simply a way to explicitly represent the empty string of vocabulary symbols, and (2) the italicized tokens are not part of the grammar, but rather stubs designed to give you a little something to do.

1. sentence → nounphrase verbphrase nounphrase
2. nounphrase → the noun
3. nounphrase → the adjectivelist noun
4. verbphrase → verb
5. adjectivelist → ϵ
6. adjectivelist → adjective adjectivelist
7. noun → *noun1*
8. noun → *noun2*
9. noun → *noun3*
10. noun → *noun4*
11. noun → *noun5*
12. verb → *verb1*
13. verb → *verb2*
14. verb → *verb3*
15. verb → *verb4*
16. verb → *verb5*
17. adjective → *adjective1*
18. adjective → *adjective2*
19. adjective → *adjective3*
20. adjective → *adjective4*
21. adjective → *adjective5*

With this partial CFG in mind, please do the following tasks, some of which call on you to think up some words, some of which require that you do derivations, and some of which require a bit of analysis.

1. Render the given partial CFG a complete CFG by filling in the five "noun slots" with five reasonable **nouns** of your own choosing, by filling in the five "verb slots" with five reasonable **past tense verbs** of your own choosing, and by filling in the five "adjective slots" with five reasonable **adjectives** of your own choosing.
2. Present a line by line derivation of a sentence in the language generated by this grammar that **does not contain any adjectives**.
3. Present a line by line derivation of a sentence in the language generated by this grammar that **contains two consecutive adjectives**.
4. Determine the number of sentences in the language generated by this grammar which do not contain any adjectives. Please give a short justification for your answer.
5. Determine the number of sentences in the language generated by this grammar which do contain at least one adjective. Please give a short justification for your answer.

Exercise: Define a CFG for positive quaternian numbers with no leading zeros

Please define a CFG for the set of *positive quaternary integers* with no leading zeros. A **quaternary** integer is a base-4 integer. It uses the digits 0, 1, 2 and 3 to represent any integer. (A leading zero is a zero that starts an integer that is of length greater than 1. Thus, both `01` and `00003` have leading zeros, but `0` does not.) For example, the following are positive quaternary integers with no leading zeros:

1. 0
2. 1
3. 2
4. 3
5. 33
6. 101
7. 123000
8. 333221122333
9. 102030
10. 100000000000000000000000000000000000

In doing this task, please write down each of the four parts of your CFG explicitly. That is, write down the terminal set, write down the nonterminal set. write down the productions, and write down the start symbol.

Exercise: Define a CFG for strings of alternating plusses and minuses

Please define a context free grammar, by presenting just the productions, for the language of all nonempty strings of alternating plusses and minuses in which you can start with either symbol.

By way of example, here are several of the smallest sentences in this language:

- `-`
- `+`
- `-+`
- `+-`
- `-+-`
- `+-+`
- `-+-+`
- `+-+-`