Parsing

The derivation of a string produces a parse tree for the string:

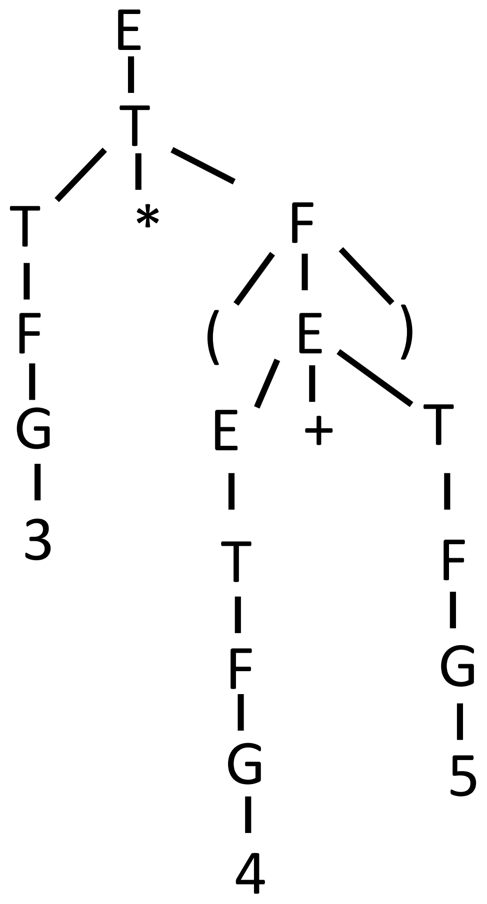
Grammar:

E => E+T | E-T | T
T => T*F | T/F | F
F => (E) | G
G => G digit | digit

Derivation:

E => T
=> T*F
=> F*F
=> G*F
=> 3*F
=> 3*(E)
=> 3*(E+T)
=> 3*(T+T)
=> 3*(F+T)
=> 3*(G+T)
=> 3*(4+T)
=> 3*(4+F)
=> 3*(4+G)
=> 3*(4+5)

Parse Tree:



Example 1: Find a grammar for $\{0^n1^n \mid n >= 0\}$ This is one of the languages we showed isn't regular.

$S => 0\ S\ 1 \mid \varepsilon$

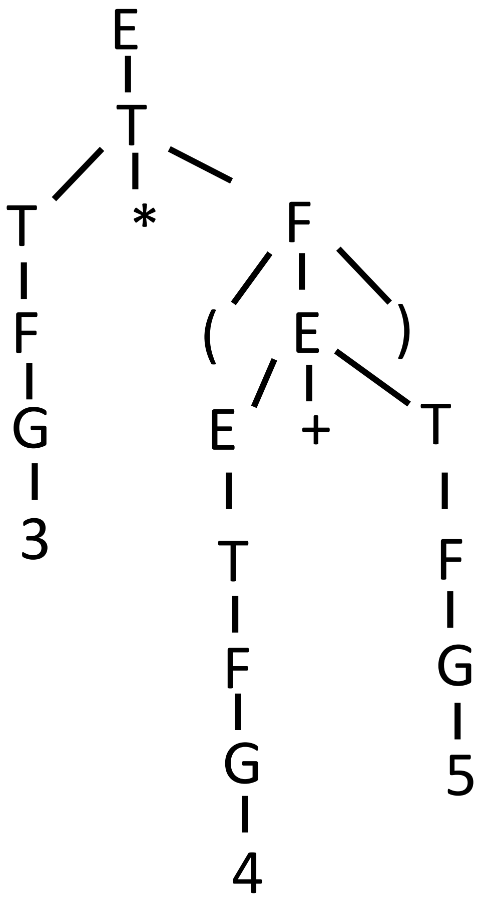
Example 2: Find a grammar for $\{ww^{rev} \mid w \in (0+1)^* \}$ (even-length palindromes)

$S => 0\ S\ 0 \mid 1\ S\ 1 \mid \varepsilon$

Example 3: Find a grammar for the language of all palindromes of 0's and 1's

$S => 0\ S\ 0 \mid 1\ S\ 1 \mid 0 \mid 1 \mid \varepsilon$

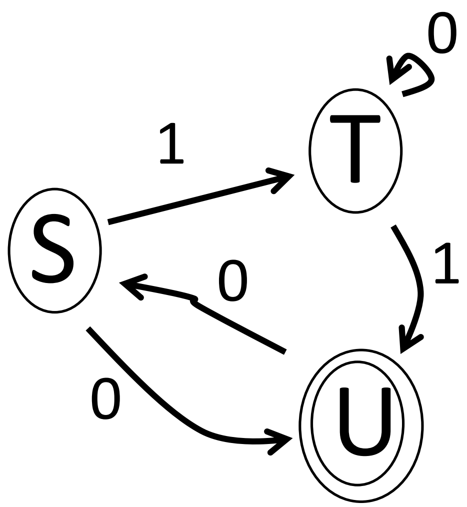
Note that we can reproduce the string being parsed with a left-to-right traversal of the leaves of the parse tree:



3*(4+5)

Regular Grammars

Consider the DFA

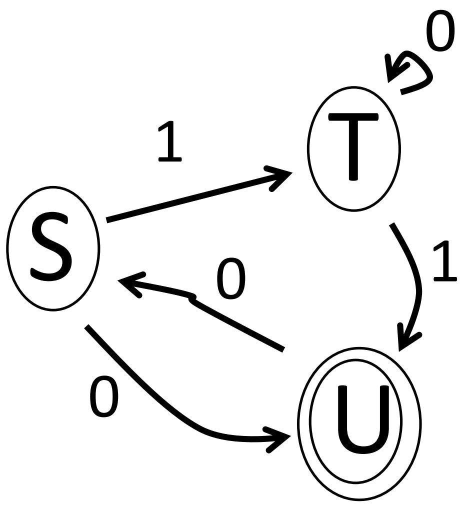


Here is a grammar for the language this accepts:

S => 1T | 0U

T => 0T | 1U

U => 0S | ε



S => 1T | 0U
T => 0T | 1U
U => 0S | ε

Here is a derivation of 00101:

S => 0<u>U</u>
=> 00<u>S</u>
=> 001<u>T</u>
=> 0010<u>T</u>
=> 00101<u>U</u>
=> 00101

Definition: A grammar that has only rules of the forms

- X => a Y
- X => ε

is called a *regular grammar*.

Theorem: The language defined by a regular grammar is regular. All regular languages are defined by regular grammars.

Proof: Given a regular grammar, build an NFA with the non-terminal symbols as states and transition function δ defined by: if X => a Y is a grammar rule then $\delta(X,a)$ includes Y. If there is a rule X => ε then X is a final state in the NFA. Note that every step of derivation with the grammar has the form $S \overset{*}{\Rightarrow} \alpha Y$ where α is a string containing only terminal symbols. An easy induction on the length of α shows that $S \overset{*}{\Rightarrow} \alpha Y$ if and only if the string α takes the automaton from its start state to state Y. This means the automaton accepts the same strings as are generated by the grammar.

On the other hand, if we start with a regular language it must have a DFA that accepts it. We can generate a regular grammar from this DFA. Again, a straightforward induction shows that the grammar defines the same language as the automaton.

Since regular grammars are context-free, we see that all regular languages are context free. But the family of context-free languages includes many languages that are not regular, including

$\{0^n1^n \mid n >= 0\}$ $\{ww^{rev} \mid w \in (0+1)^*\}$