Grammars II

See Section 5.2
The derivation of a string produces a parse tree for the string:

**Grammar:**

\[
\begin{align*}
E & \rightarrow E + T \mid E - T \mid T \\
T & \rightarrow T * F \mid T / F \mid F \\
F & \rightarrow (E) \mid G \\
G & \rightarrow G \text{ digit} \mid \text{ digit}
\end{align*}
\]

**Derivation:**

\[
\begin{align*}
E & \rightarrow T \\
& \rightarrow T * F \\
& \rightarrow F * F \\
& \rightarrow G * F \\
& \rightarrow 3 * F \\
& \rightarrow 3 * (E) \\
& \rightarrow 3 * (E + T) \\
& \rightarrow 3 * (T + T) \\
& \rightarrow 3 * (F + T) \\
& \rightarrow 3 * (G + T) \\
& \rightarrow 3 * (4 + T) \\
& \rightarrow 3 * (4 + F) \\
& \rightarrow 3 * (4 + G) \\
& \rightarrow 3 * (4 + 5)
\end{align*}
\]

**Parse Tree:**

![Parse Tree](image-url)
Example 1: Find a grammar for \( \{0^n1^n \mid n \geq 0\} \) This is one of the languages we showed isn't regular.
\[
S \Rightarrow 0\, S\, 1 \mid \varepsilon
\]

Example 2: Find a grammar for \( \{0^{2^n}1^n \mid n, m \geq 0\} \)
\[
s \Rightarrow 0\, S\, 1 \mid T \\
T \Rightarrow 2\, T \mid \varepsilon
\]

Example 3: Find a grammar for \( \{ww^{\text{rev}} \mid w \in (0+1)^*\} \) (even-length palindromes)
\[
S \Rightarrow 0\, S\, 0 \mid 1\, S\, 1 \mid \varepsilon
\]

Example 4: Find a grammar for the language of all palindromes of 0's and 1's
\[
S \Rightarrow 0\, S\, 0 \mid 1\, S\, 1 \mid 0 \mid 1 \mid \varepsilon
\]
Note that we can reproduce the string being parsed with a left-to-right traversal of the leaves of the parse tree:

\[ 3 \times (4 + 5) \]
Regular Grammars

Consider the DFA

Here is a grammar for the language this accepts:

\[ S \rightarrow 1T \mid 0U \]
\[ T \rightarrow 0T \mid 1U \]
\[ U \rightarrow 0S \mid \varepsilon \]
Here is a derivation of 00101:

\[ S \Rightarrow 0U \]

\[ \Rightarrow 00S \]

\[ \Rightarrow 001T \]

\[ \Rightarrow 0010T \]

\[ \Rightarrow 00101U \]

\[ \Rightarrow 00101 \]
Definition: A grammar that has only rules of the forms
• X => a Y
• X => a
is called a regular grammar.

For example, here is a regular grammar:

S => 0S | 1T | 0
T => 0T | 1S | 1

A typical derivation is S => 0S => 00S => 001T => 0010T => 00101
**Theorem:** The language defined by a regular grammar is regular.

**Proof:** Given a regular grammar, build an NFA from it. The states of the NFA are the non-terminal symbols of the grammar, plus an extra final state called "Accept". If $X \Rightarrow aY$ is a rule in the grammar add a transition in the NFA $\delta(X,a) = Y$. If $X \Rightarrow a$ is a grammar rule make a transition $\delta(X,a) = \text{Accept}$.

Every step except the last of a derivation of a string in the regular grammar has the form $S \Rightarrow^{*} \alpha X$. An easy induction shows that $S \Rightarrow^{*} \alpha X$ if and only if string $\alpha$ takes the NFA from state $S$ to state $X$. The grammar derives string $w$ if and only if $w = \alpha a$ and there is a non-terminal symbol $X$ with $S \Rightarrow^{*} \alpha X$ and $X \Rightarrow a$, which says the string $\alpha a$ will take the NFA to state Accept. This says the grammar derives string $w$ if and only if the NFA accepts the string $w$. 
For example, start with the regular grammar

\[ S \rightarrow 0S \mid 1T \mid 0 \]
\[ T \rightarrow 0T \mid 1S \mid 1 \]

This gives the NFA

![NFA Diagram]

Both the grammar and the NFA describe strings with an even number of 1's.
**Theorem**: Every regular language has a regular grammar.

**Proof**: Start with DFA that describes a regular language. We will build a grammar for the language. The non-terminal symbols of the grammar will be the names of the states of the DFA. If the DFA has transition $\delta(X,a) = Y$, add the grammar rule $X \Rightarrow aY$. If the DFA has transition $\delta(X,a) = Y$ and $Y$ is a final state, also add the grammar rule $X \Rightarrow a$. A string $w = \alpha a$ is accepted by the DFA if and only if $S \overset{*}{\Rightarrow} \alpha X$ and $X \Rightarrow a$, so $w$ is accepted by the DFA if and only if $S \overset{*}{\Rightarrow} w$. 
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
\[ \{0^n1^n \mid n \geq 0\} \text{ and } \{ww^{rev} \mid w \in (0+1)^*\} \]