This lecture is about representing English syntax using parse trees. Parse trees go beyond part-of-speech tags, aiming to represent more fully the syntactic/grammatical structure of a sentence. The most common type of parse tree is made out of **constituents** (sometimes called **phrases**), which are contiguous (in English, usually) sequences of words that behave as a unit.

A context-free grammar G is a tuple (Σ, N, S, R). Σ is a vocabulary of terminal symbols. N is a set of nonterminal symbols (also called variables), and $S \in N$ is a special "start" symbol. R is a set of **production rules** of the form $X \rightarrow \alpha$ where $X \in N$ (a nonterminal symbol) and $\alpha \in (N \cup \Sigma)^*$ (a sequence of terminals and nonterminals). A **derivation** of G is obtained by a nondeterministic (in general) process:

- Write down the start symbol S.
- While there are any nonterminal symbols written down:
  - Choose one nondeterministically. Let X denote that symbol.
  - Choose a production rule of the form $X \rightarrow \alpha$.
  - Erase the X and write α in its place.

You can think of a parse tree as a visualization of the production rules used to derive a grammatical sentence using a context-free grammar. **Parsing** (in NLP) most commonly refers to the problem of *recovering* the parse tree of a sentence.

In the lecture you saw some examples of rules that might be used to represent English syntax. You'll see more in the reading, as well as some parse trees. You'll get to try your hand at this in the next assignment. We can get a really long way using context-free production rules, but there are some challenges:

- Some phenomena, like agreement, don't fit into context-free rules very elegantly.
- If we want broad coverage, there are an awful lot of rules!
- It's very hard to write rules that only allow correct parses without allowing bad ones.

We also talked about an alternative representation to phrase-structure trees, **dependency trees**, which focus on the grammatical relationships between the words. Under certain assumptions (that mostly hold in English), we can view dependency trees as derivations of context-free grammars that use words as both terminals and nonterminals. Again, examples of dependency trees can be found in chapter 12.