Lecture 12

Moving Beyond Context-Free Grammars

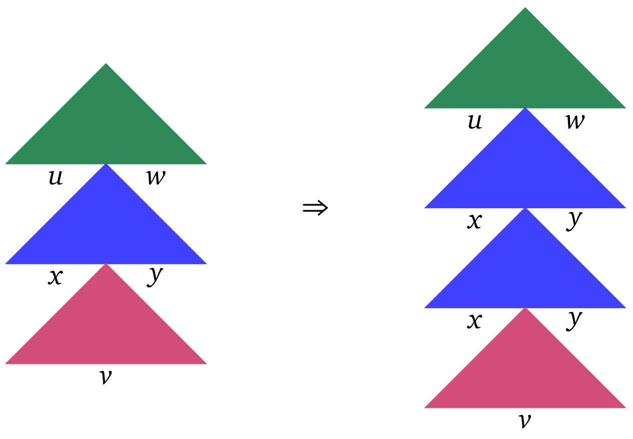
1 Not All Natural Languages are Context-Free

1.1 Context-Free Pumping Lemma

Theorem 12.1 (CFL Pumping Lemma). If L is a context-free string language, then there is some constant $k \geq 0$ such that

- $uxvyw \in L$, and
- $|uxvyw| \geq k$

jointly imply $ux^n vy^n w \in L$ for all $n \geq 1$. $\square$



Lemma 12.2. None of the following languages are context-free:

- $a_1^n a_2^n \cdots a_i^n$, for any fixed choice of $i \geq 3$.
- $\{ww \mid w \in \Sigma^*, |\Sigma| \geq 2\}$,
- a^{2^n} $\square$

1.2 Mildly Context-Sensitive Languages

Definition 12.3 (Mild Context-Sensitivity). A class of string languages is *mildly context-sensitive* iff

1. it properly includes the class of context-free languages,
2. each language can be generated by a grammar that allows for parsing in polynomial time,
3. each language is of constant growth,
4. each language can be generated by a grammar with only a bounded number of crossing dependencies.

Exercise 12.1. Show that $a^n b^n c^n$ is of constant growth, whereas a^{2^n} is not. ⊙

This definition is not particularly elegant as it puts restrictions on the languages as well as the grammars that generate them. A more unified definition in terms of just one of the two would be more appealing. In addition, the clause about crossing dependencies is very vague. Fortunately it can be made more precise via the concept of *semilinearity*.

1.3 Semilinear Languages

In order to define semilinearity, we first need to introduce the *Parikh map*. Intuitively, the Parikh map counts for each symbol of the alphabet how often it occurs in a given string. This is modeled mathematically as a function that takes a string as input and returns a vector of integers — the *Parikh vector* — such that the ith integer is the number of occurrences of the ith symbol of the alphabet (so we have to posit some arbitrary linear order for our alphabet symbols).

Definition 12.4 (Vector addition). Let $\vec{u} := \langle u_1, \ldots, u_k \rangle$ and $\vec{v} := \langle v_1, \ldots, v_k \rangle$ be vectors in $\mathbb{N}^k$. Then their sum $\vec{u} + \vec{v}$ is the vector $\langle u_1 + v_1, u_2 + v_2, \ldots, u_k + v_k \rangle$.

Definition 12.5 (Parikh Map). The *Parikh vector* of a string is the function $p : \Sigma^* \to \mathbb{N}^{|\Sigma|}$ such that

$$p(a_1 a_2 \cdots a_n) := \begin{cases} 0^{i-1} \cdot \langle 1 \rangle \cdot 0^{k-i} & \text{if } n = 1 \text{ and } a_1 \text{ is the } i\text{-th symbol of } \Sigma, \\ p(a_1) + p(a_2 \cdots a_n) & \text{otherwise} \end{cases}$$

The *Parikh map* $\hat{p}$ associates a language L with the set $\{p(w) \mid w \in L\}$. Two languages L and L' are *letter equivalent* iff $\hat{p}(L) = \hat{p}(L')$.

Example 12.1 Parikh vectors and maps

Consider the string language $\{a, b\}^*$, which contains all possible strings over a and b. The Parikh map computes the following Parikh vectors for this language.

String	**Parikh Vector**
ε	$\langle 0,0\rangle$
a	$\langle 1,0\rangle$
b	$\langle 0,1\rangle$
aa	$\langle 2,0\rangle$
ab	$\langle 1,1\rangle$
ba	$\langle 1,1\rangle$
bb	$\langle 0,2\rangle$
$\vdots$	$\vdots$

The set of all Parikh vectors is $\{\langle m,n\rangle \mid m,n \in \mathbb{N}\}$, which is identical to $\mathbb{N}\times\mathbb{N}$. In a certain sense, then, we can think of $\{a,b\}^*$ as the set of all vectors in the first quadrant of the Cartesian plane. This is a simplification, though, since $\{a,b\}^*$ contains many strings that map to the same Parikh vector because the Parikh map completely ignores linear order.

Exercise 12.2. Define a string language that is distinct from $\{a,b\}^*$ but letter-equivalent to it. $\odot$

Exercise 12.3. Define a string language as in the previous exercise, with the additional requirement that no proper subset of that language is letter equivalent to $\{a,b\}^*$. $\odot$

The set $\mathbb{N}\times\mathbb{N}$ is obviously infinite, but nonetheless it can be generated from a finite number of vectors. Any vector of that set is a directed arrow in the first quadrant of that Cartesian plane. Wherever that vector points, we can get to that point by only moving up and to the right. For example, the location pointed to by the vector $(2,1)$ can be reached by starting at $(0,0)$ and taking two steps to the right, immediately followed by one step up. This can be written as $(2,1) = (0,0)+(1,0)+(1,0)+(0,1) = (0,0)+2\cdot(1,0)+1\cdot(0,1)$. Every vector of $\mathbb{N}\times\mathbb{N}$ can be described in this fashion, we just have to change the multipliers accordingly. Consequently, $\mathbb{N}\times\mathbb{N}$ is equivalent to the set $\{(0,0)+t_1\cdot(1,0)+t_2\cdot(0,1) \mid t_1,t_2 \in \mathbb{N}\}$. Sets of this form are called *linear*.

Definition 12.6 (Linear Set). A set $S \subseteq \mathbb{N}^k$ is *linear* iff it is of the form $\{u_0 + t_1u_1 + \cdots + t_nu_n | t_1,\ldots,t_n \in \mathbb{N}\}$ for some fixed base vectors $u_0,u_1,\ldots,u_n$, $n \geq 0$.

Exercise 12.4. Show that the Parikh map yields a linear set for $a^nb^nc^n$, $n \geq 1$. $\odot$

Many interesting languages are not linear under the Parikh map. However, they are still semilinear

Definition 12.7 (Semilinearity). A set is *semilinear* iff it is the union of a finite number of linear sets. A language is semilinear iff its image under the Parikh map is semilinear.

1.4 Semilinearity and Constant Growth

Theorem 12.8. Every semilinear set is of constant-growth. □

Corollary 12.9. a^{2^n} is not semilinear. □

Exercise 12.5. Give an example of a language that is of constant-growth but not semilinear. ⊙

1.5 Semilinearity = Regular Backbone + String Permutation

Theorem 12.10 (Parikh's Theorem). For every semilinear language L there is a regular language R such that $\hat{p}(L) = \hat{p}(R)$. □

Exercise 12.6. The inverse of Parikh's Theorem also holds: for every regular language R there is a semilinear language L such that $\hat{p}(L) = \hat{p}(R)$. Explain why this is (trivially) the case. ⊙

2 Minimalist Grammars