Formal Languages

This page introduces the key ingredients used to define formal languages.

1. An alphabet $\Sigma$. $\Sigma$ is a finite set that contains all relevant symbols.
   - In the context of English strings, $\Sigma = \{a, b, c, \ldots, z\}$ (this is indeed the English alphabet we know of)
   - In the context of binary strings, $\Sigma = \{0, 1\}$
   - In the context of phone numbers, $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

2. A string is a finite sequence of symbols in the alphabet.
   - If $\Sigma = \{a, b, c, \ldots, z\}$:
     - discrete, math, and qpcuyzmg are all strings, regardless of whether we recognize them (you probably won't recognize the last word).
     - discrete and cs230 are not strings because they use symbols not in the alphabet (despite that you can probably recognize both words).
   - If $\Sigma = \{0, 1\}$:
     - 000001, 111, 1, 10110 are all strings.
     - 1111111111 is not a string because it is not finite.
   - If $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$:
     - 9191234567 is a string.
     - 01234567890123456789 is also a string. You know it does not represent anyone's phone numbers, but we have not yet formalized what makes valid phone numbers yet.
   - The size of a string is the length of the sequence.
   - We use the symbol $\lambda$ (or $\varepsilon$) to represent the empty string. $\lambda$ (or $\varepsilon$) has size 0 and it is a valid string for every alphabet.

3. A language is a (not necessarily finite) set of strings over the alphabet $\Sigma$.
   - If $\Sigma = \{a, b, c, \ldots, z\}$:
     - \{discrete, math\} is a language (that contains only the two words).
     - $\emptyset$ is a language (that contains no words).
     - You can try to define the set of all valid English words this way (but other people may not agree with your version).
   - If $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$:
     - We can list all 10-digit phone numbers one-by-one. That is a set of $10^{10} = 10,000,000,000$ strings.
There must be a better way of describing the set of 10-digit phone numbers, you say. Indeed there is. Here are some useful notations:

- The **concatenation** of two strings \( u \) and \( v \) can be written as just \( uv \).
  - For example, if \( u = \text{disc} \) and \( v = \text{rete} \), then \( uv = \text{discrete} \). Note how we use bold fonts for strings and italic fonts for symbols.
  - It can also be written as \( u \circ v \). Do not conflate with function composition.
  - \( u \circ \lambda = \lambda \circ u = u \).

- We can simply use the cardinality notation to represent the size of a string.
  - \( |uv| = 8 \).

- \( v^k \) represents the concatenation of \( k \) identical copies of \( v \).
  - \( v^2 = \text{reterete} \).
  - \( v^0 = \lambda \).
  - \( (uv)^2 = \text{discretediscrete} \).

- \( \Sigma^* \) represents the set of all strings using symbols in \( \Sigma \), while \( \Sigma^+ \) represents the set of all **nonempty** strings using symbols in \( \Sigma \).
  - \( \Sigma^+ = \Sigma^* \setminus \{\lambda\} \).

- If \( L \) is a language, \( L^k \) represents the concatenation of \( k \) (not necessarily identical) strings in \( L \).
  - If \( L = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \), then \( L^{10} \) is the set of 10-digit phone numbers.

- \( L^* \) represents the concatenation of zero or more strings in \( L \), while \( L^+ \) represents the concatenation of one or more strings in \( L \).
  - \( L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \ldots \)
  - \( L^+ = L^1 \cup L^2 \cup L^3 \ldots \)
  - \( L^+ = L^* \setminus L^0 = L^* \setminus \{\lambda\} \)
  - As you see, since languages are sets, all set operators can be used on languages.