

## Stop word removal as a function

### PREREQUISITES

- functions (domain terminology)

This unit illustrates how one might define stop word removal as a mathematical function *del* (read *delete*).

First, we fix some alphabet  $\Sigma$  and let  $S$  be some finite set of symbols drawn from  $\Sigma$ . For every such  $S$ , we define a deletion function  $del_S$  that maps strings over  $\Sigma$  to strings over  $\Sigma - S$ . In mathematical notation,  $del_S : \Sigma^* \rightarrow (\Sigma - S)^*$ .

This only tells us the domain and co-domain of  $del_S$ , but not how exactly inputs and outputs are connected to each other. For any string of the form  $u_1 \cdots u_n$  (where  $n \geq 0$  and each  $u_i$  is a symbol drawn from  $\Sigma$ ), we define

$$del_S(u_1 \cdots u_n) := \begin{cases} \varepsilon & \text{if } u_1 \cdots u_n = \varepsilon \\ del_S(u_2 \cdots u_n) & \text{if } u_1 \in S \\ u_1 \cdot del_S(u_2 \cdots u_n) & \text{otherwise} \end{cases}$$

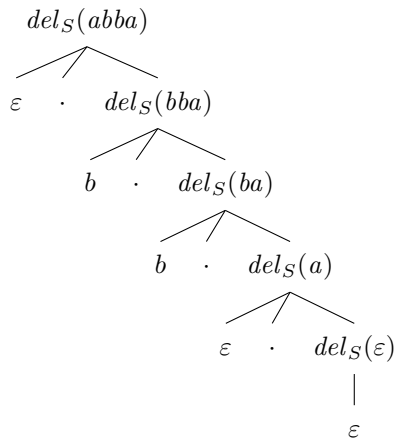
### EXAMPLE 1.

Suppose  $\Sigma := \{a, b\}$  and  $S := \{a\}$ . Let  $s := abba$ . Then  $del_S(s)$  should yield  $bb$ . To this end, we compute  $del_S(s)$  in a stepwise fashion:

$$\begin{aligned} del_S(s) &= del_S(abba) \\ &= del_S(bba) \\ &= b \cdot del_S(ba) \\ &= b \cdot b \cdot del_S(a) \\ &= b \cdot b \cdot del_S(\varepsilon) \\ &= b \cdot b \cdot \varepsilon \\ &= b \cdot b \\ &= bb \end{aligned}$$

So  $del_S(abba) = bb$ , as expected.

As you can see,  $del_S$  is partially defined in terms of itself: the value of  $del_S(abba)$  is inferred from the value of  $del_S(bba)$ . This is called a **recursive** definition. We can visualize the computation of this recursive function as below:



Every recursive function has one or more **base cases** and a **recursion step**. The base cases are those where the value of the function can be determined without recursion. For  $del$ , there is only the base case  $del_S(\epsilon) = \epsilon$ . Notice how in the graph above  $del_S(\epsilon)$  does not contain any further instances of  $del_S$ . Instead, we immediately get  $\epsilon$  as the output. The recursion step defines the function in terms of the function itself. In the graph above, that's every instance of  $del_S$  which has another instance of  $del_S$  below it.

## EXERCISE 1.

Here is another recursively defined function.

$$f(x, y) := \begin{cases} x & \text{if } y \leq 1 \\ x + f(x, y - 1) & \text{otherwise} \end{cases}$$

What does this function do? Is there a commonly used name for it?

## EXERCISE 2.

This continues the previous exercise. Draw a diagram like the one above for  $f(5, 4)$ .

## EXERCISE 3.

Give a recursive definition of a function that takes two arguments: a string  $u := u_1 \cdots u_n$  over alphabet  $\Sigma$ , and a set  $S$  of symbols drawn from  $\Sigma$ . The function returns 1 if at least one member of  $S$  occurs in  $u$ , and 0 otherwise.

## EXERCISE 4.

This continues the previous exercise. Draw a diagram like the one above for

## Stop word removal as a function

---

$f(aaba, \{b\})$  and  $f(aaba, \{c, d, e\})$ .