Trees and structural ambiguity
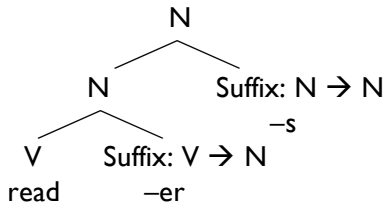
We've seen how we can use tree diagrams to illustrate the structure of morphologically complex words.

```
                    N
             ┌──────┴──────┐
             N          Suffix: N → N
         ┌───┴───┐          –s
         V   Suffix: V → N
        read      –er
```

When we look at trees that contain suffixes as well as prefixes, it's not always immediately clear how their structure should look like.

```
          ┌─────────┐                              ┌────────┐
     ┌────┴───┐   –able          ?            re–  ┌───┴────┐

    re–     seal                                  seal    –able
```
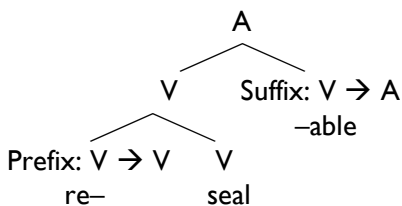
Only when we look at the properties of the pre- and suffixes involved, we can decide which of the two structures must be the right one:

We know that *seal* is a verb (I want to <u>seal</u> (this deal…); seal – sealed), *re–* combines with verbs to create verbs, and *–able* attaches to verbs to create adjectives (for details see notes from last class).

Lets focus on the structure on the left first: Here *seal* (V) combines with *re–* (V → V), to form the verb *reseal*. *–able* (V → A) then combines with *reseal* to form the adjective *resealable*. From what we've found out about *re–* and *–able* so far, nothing's wrong with this tree.

Now lets compare that to the structure on the right: Here *seal* (V) combines with *–able* (V → A) to form the adjective *sealable*. Now however, we can no longer attach *re–* since we know that *re–* does not attach to adjectives. This rules out the structure on the right. We've successfully argued that the structure on the left must be the right one.
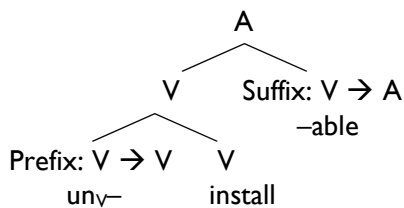
```
                    A
             ┌──────┴──────┐
             V          Suffix: V → A
         ┌───┴───┐          –able
  Prefix: V → V   V
       re–       seal
```

Lets look at another case: *uninstallable*.

```
        /\                              /\
       /\  −able          ?         un−   /\
      /  \                |              /  \
    un−   install                  install  −able
```
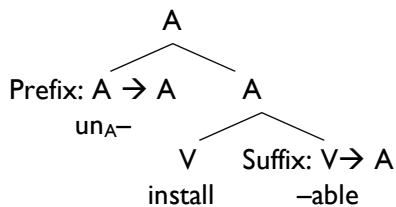
On your homework, you found out that *un−* has a double life. Either it attaches to adjectives, to create new adjectives (happy – unhappy, true – untrue, exciting – unexciting, …), or it attaches to verbs to create verbs (block – unblock, install – uninstall, do – undo, …). To distinguish them, lets call the first *un$_A$−* and the second one *un$_V$−*. How does that play out for the two structures we investigate:

On the left, *un−* attaches to the verb *install*, the result of this is the verb *uninstall*. *−able* then attaches to this verb to form the adjective *uninstallable*.

```
                    A
                   / \
                  V   Suffix: V → A
                 / \      −able
    Prefix: V → V   V
        un$_V$−      install
```

But the structure on the right works out as well: *−able* attaches to the verb *install* to form the adjective *installable*. The form of *un−* that attaches to adjectives can then attach to *installable* to form the adjective *uninstallable*.

```
              A
             / \
    Prefix: A → A   A
        un$_A$−    / \
                  V   Suffix: V → A
                install   −able
```

We then found out that we can tell difference between the two structures, namely in their meaning. For the first structure we predict the meaning "can be uninstalled", and this is in fact how you can use the word, for instance in a scenario like this: "Don't worry about trying out the software. You can remove it later, it's easily uninstallable."

For the second structure, we predict a different meaning. It can be paraphrased as "not installable". And we can use the word in this sense too: "Your computer is really old. Sorry, but the software is uninstallable on this machine."

We can also detect the two slightly different meanings of un− here. While un$_A$− has the meaning 'not A', where A is the meaning of the adjective, un$_V$− talks about reversing some Ved state: to

uninstall something, it must have been installed before, to unblock something, it must be in a blocked state, etc.

So here we have a case where both structures are possible for the word we see. The two structures predict different meanings, and in fact, that is what we see!