Combining *n*-gram grammars (Solutions)

Exercise 1.
Write two negative grammars such that the first only generates *ab* and *ba*, whereas the latter generates all strings of the form *ab*, *aab*, *aaab*, and so on. Then build the corresponding combined grammar. What is the language generated by the combined grammar? Is this the correct result?

Solution

1. $G_1 := \{\rtimes\rtimes\ltimes, \rtimes a\ltimes, \rtimes b\ltimes, \rtimes aa, \rtimes bb, aa\ltimes, bb\ltimes, aaa, aab, aba, abb, baa, bab, bba, bbb\}$
2. $G_2 := \{\rtimes\ltimes, \rtimes b, ba, bb, a\ltimes\}$
3. $G_1 \cup G_2 := \{\rtimes \rtimes \ltimes, \rtimes a\ltimes, \rtimes b\ltimes, \rtimes aa, \rtimes bb, aa\ltimes, bb\ltimes,$
   $aaa, aab, aba, abb, baa, bab, bba, bbb,$
   $\rtimes \ltimes, \rtimes b, ba, bb, a\ltimes\}$
4. $L(G_1 \cup G_2)$ contains only the string *ab*, which is the correct result as $L(G_1) \cap L(G_2) = \{ab\}$.

Explanation

This exercise is slightly cumbersome because it asks you to provide negative grammars when positive grammars would be much easier. But we can start out with negative grammars and then convert them to negative ones.

The first language contains both *ab* and *ba*, but not *aba*. This means that our grammar cannot be a bigram grammar: all bigrams of the illicit string *aba* are bigrams of the licit strings *ab* and *ba*, so it's impossible for a bigram grammar to generate both *ab* and *ba* but not *aba*. So let us take the step up to trigrams. Now it is very easy to write a positive grammar that generates only those two strings. The grammar must allow all of the following trigrams: $\rtimes \rtimes a$, $\rtimes \rtimes b$, $\rtimes ab$, $\rtimes ba$, $ab\ltimes$, $ba\ltimes$, $a \ltimes \ltimes$, $b \ltimes \ltimes$. We can then convert that into a negative grammar by constructing $\{a,b\}^3_E$ and keeping only those trigrams that are not already in our positive grammar. And that gives us the grammar in the solution above (except that we do not list $\rtimes \ltimes \ltimes$ because that is redundant if the grammar already contains $\rtimes \rtimes \ltimes$).

Now we do the same for the second language. Here things are easier as we can actually make do with a bigram grammar: all strings must start with *a*, *b* cannot occur anywhere except at the end of the string, and *a* may not occur at the end of the string. This is fairly easy to express directly as aa negative bigram grammar: $\rtimes\ltimes$ and $\rtimes b$ ("only start with *a*"), *ba* and *bb* ("*b* can only occur at the end of the string"), and $a\ltimes$ ("no *a* at the end of the string").

When we combine the two grammars, we take their union. Since one of them is a trigram grammar and the other is a bigram grammar, the result will be a mixed grammar where not all *n*-grams have the same length. We could convert our bigram grammar to a trigram grammar first, but that's just extra work with no gain and does not affect anything here. Hence the combination of the two grammars is a mixed negative grammar. Due to the bigrams it contains, it cannot generate any strings that do not start with a or do not end in b. Due to the trigrams it contains, it cannot generate any strings of length 3 or more. But the only string of length 2 or less that starts with a and ends in b is ab. This is exactly the intersection of $L(G_1) := \{ab, ba\}$ with $L(G_2)$, which contains the strings ab, aab, $aaab$, and so on.

(A brief aside: based on the discussion above, we can infer that $G_1 \cup G_2$ isn't the smallest negative grammar that generates $L(G_1 \cup G_2)$, a smaller option would be the bigram grammar that contains all of the following forbidden bigrams: $\rtimes\ltimes$, $\rtimes b$, aa, ba, bb, $a\ltimes$.)

Exercise 2.
Compute $A \cap B$ in the same fashion.

Solution

According to De Morgan's Law, $A \cap B = \overline{\overline{A} \cup \overline{B}}$

- $\overline{A} = U - A = \{a, b, c, d, e\} - \{a, b, c\} = \{d, e\}$
- $\overline{B} = U - B = \{a, b, c, d, e\} - \{b, c, d\} = \{a, e\}$
- $\overline{A} \cup \overline{B} = \{a, d, e\}$
- $\overline{\overline{A} \cup \overline{B}} = U - (\overline{A} \cup \overline{B}) = \{a, b, c, d, e\} - \{a, d, e\} = \{b, c\}$

Exercise 3.
Suppose ${}^{+}G$ is a positive grammar containing the bigrams $\rtimes a$, aa, and $a\ltimes$. Assume furthermore that all symbols must be a. Compute $\overline{G}$, then verify for yourself that ${}^{-}\overline{G}$ is the negative grammar that accepts the same strings as ${}^{+}G$.

Solution

Since all symbols must be a, Σ_E^2 contains all of the following:

1. $\rtimes\ltimes$
2. $\rtimes a$
3. aa
4. $a\ltimes$

For simplicity, we omit useless bigrams: $\ltimes\rtimes$, $a\rtimes$, and $\ltimes a$. Hence $\overline{G} = \Sigma_E^2 - +^G = \rtimes\ltimes$. The negative grammar containing only $\rtimes\ltimes$ generates all strings over a except the empty string, i.e. all strings over a that contain 1 or more instances of a. But this is exactly the language generated by the positive grammar that contains $\rtimes a$, aa, and $a\ltimes$.