Computational Linguistics I, Winter 2006. Marcus Kracht

To be submitted: Friday, March 10, 2006.

Propositional formulae are written in many different ways. The most common one is to write unary symbols as prefixes, and binary symbols as infixes. There are also brackets to help disambiguate. Here is a different way to write formulae. Instead of writing `p→q` we write `→p q`, instead of `p∨q` we write `∨pq`. However, we use no brackets at all. This is known as Polish notation.

[A 7.1] Write down the CF grammar rules for the natural notation and the Polish Notation.

[A 7.2] Give the symbols of the language the following weights: variables count -1, negation and unary symbols in general count 0, and binary symbols count 1. For each subword, the weight of the subword is the sum of all weights of the occurring symbols. Now show the following: if a string is in Polish Notation, then all its prefixes have nonnegative weight, while the string itself has weight -1. This characterisation is exact: any string that has this property is a string in Polish Notation!

[A 7.3] Choose an encoding of the language as follows: lower case letters are for variables, 'N' is for $\neg$ (negation), 'C' for $\wedge$ (conjunction), 'D' for $\vee$ (disjunction), and 'I' for $\rightarrow$ (implication). Write a program that checks whether a string is in Polish Form (using the given proxies for the symbols).

[A 7.4] Write a program that returns for a term in Polish Notation its immediate subterms. Do the same for a bracketed string. *Hint.* You have to choose the function carefully, since there may be either 0, 1 or 2 subterms, so you cannot just make the result a string.