Functional Approach to Natural Language Processing

Ján Kollár, Milan Spišiak and Michal Sičák

The evolution of natural language depends on both interpersonal communication and internal thinking. In our symbolic and grammatical approach, percepted real world objects are symbolized and transformed to internal symbols being subjects of reasoning as concepts in human an/or machine mind.

We present an approximative evaluation of communication speed in human brain as well as the results of our experiment with the aquisition of voice symbols of natural language. As we will see, symbolic approach is quite realistic, provided that a human-like representation of memory and symbols will be developed for machines. Using functional theory of lambda calculus and combinators, we concentrate in this paper on the principles of the dynamically evolved memory. This memory is no data store, but rather complex and active supercombinator, containing no redundant parts.

Next, we use grammatical approach to communicated visual objects recognition. Although this task is not trivial and has many problematic properties, we suppose the grammar abstraction allows us to get a simpler method for the object recognition. First, we need to figure out how to describe the objects and then we can apply the method of abstraction to these data. Therefore, we primarily focus on 3D object description by grammar methods in this paper. In the grammar theory, this step is called symbolization. The symbolization ensures an object description and provides the fundamental data abstract layer. As we can see, data abstraction using functional language allows us to abstract and to process objects easily.

Applicative approach can be used in language processing even when we use context-free grammars. We show an algorithm that is able to transform any context free grammar into supercombinator form. The resulting form depends on the form of an input grammar, therefore a new problem arises: finding the proper grammar for the task at hand. We briefly show the resulting supercombinator forms of various grammar types and compare their properties. We also compare the algorithm efficiency in presented grammar cases and show, that our algorithm can be improved in case we use grammars without any cycles. We also discuss the resulting supercombinator forms in term of grammar compression and re-usability of elements, that are the end result of processing larger scale texts as an input samples.