Automatically extracting the correspondence between the natural language and the pseudo-code descriptions of instruction set manuals

2010432 Nguyen Thi Hai Yen

Instruction set architecture defines a CPU and its execution. When an automated assistant tool is considered for binary code, the formal semantics of an instruction set is required as the fundamental basis. However, often the instruction set is large, which requires heavy engineering efforts on specifying the formal semantics. For example, Intel x86 is a CISC architecture and has several thousand instructions. ARM (Advanced RISC Machine) is a RISC architecture that has few hundred instructions, but it has Cortex-A, Cortex-M, and Cortex-R series and each has 10 to 20 variations of chipsets. To overcome such situation, one possibility is to automatically extract formal semantics from an instruction set manual, written in English. For instance, x86 has Intel developer's manual, and each chipset of ARM has a reference manual, which are open to the public.

However, the interpretation of English description to a formal description, e.g., programming language, is not easy to obtain, since there are no explicit data set to train AI-related methods. Fortunately, often an instruction set manual has both English and pseudo-code descriptions for specifying the same execution step of instruction.

This thesis proposes how to automatically find the correspondence between English and pseudo-code descriptions in the instruction set manual, which will be the first step to automatically obtain interpretation rules from English to a programming language. We first limit ourselves to data processing and the load/store instructions since they are quite uniform and cover 90 % of instructions.

We first parse the English and the pseudo-code descriptions. For English sentences, we use the Stanford parser. For pseudo-code descriptions, we design 48 grammar rules for ANTLR. Next, we remove explanations in English and default declarations in pseudo-code to extract essentially describing the operation and the flag update. Lastly, we find the correspondence between the extracted parts of English and pseudo-code descriptions.

We mostly work on ARM and collect 2475 instruction descriptions over 39 chipsets, in which 2251 instructions belong to either the data processing or the load/store instruction groups. Among them, we randomly select the 30 results of the detected correspondence and examine them manually. As far as our selection, our method correctly extracts the correspondence.