Abstract

When developing programs it is of great practical interest to verify that the resulting programs have the desired properties. One of the most fundamental properties of programs is termination, i.e., that the program will not run forever, but compute some result in finitely many computation steps. The corresponding decision problem in computer science is the **halting problem**, i.e., given a description of a program and an input, decide whether the program terminates after finitely many steps or runs forever on that input.

Unfortunately, Turing showed this problem to be undecidable in general. Nevertheless, a huge number of analysis techniques which can automatically prove termination for many pairs of programs and inputs have been developed during the last decades. Nowadays, there are fully-automated tools that try to prove termination of a given program w.r.t. a given class of inputs.

However, most approaches for proving termination of programs are restricted to artificial programming languages having a comparatively simple mathematical definition and cannot handle essential features of programming languages used for real applications where exact mathematical definitions are very complex. This is especially true for logic programming, where most techniques for termination analysis are restricted to definite logic programs in contrast to real applications mostly written in the programming language **Prolog**, the main language for expert systems and applications from the artificial intelligence domain.

In this thesis, we extend the only existing approach known to be capable of handling logic programs with cuts to cover most of the features of real **Prolog** applications.

The contributions developed in this thesis are implemented in our fully automated termination prover **AProVE**. **AProVE** has reached the highest score for logic programming at the annual international Termination Competition, where the leading automated tools try to analyze termination of programs from different areas of computer science, in all years since 2004. In 2009, **AProVE** also was the only tool capable of successfully analyzing logic programs with cuts. The significance of our results is demonstrated by the empirical improvement **AProVE** shows on real **Prolog** applications used in the Termination Competition.