Logic Programming in non-conventional environments

Stefano Germano

Abstract

Logic Programming became a very useful paradigm in many different areas and thus several languages (and solvers) have been created to support various kinds of reasoning tasks. However, in the last decades, thanks also to results in the Computational Complexity area, the weaknesses and the limits of this formalism have been revealed. Therefore, we decided to study solutions that would allow the use of the *Logic Programming* paradigm in contexts, such as *Stream Reasoning*, *Big Data* or Games' *AI*, that have very specific constraints that make the practical use of logic-based formalisms not so straightforward.

Logic Programming is best used for problems where a properly defined search space can be identified and has to be explored in full. For this reason, almost all the approaches that have been tried so far, have focused on problems where the amount of input data is not huge and is stored in a few well-defined sources, which are often completely available at the beginning of the computation.
Some interesting ideas and approaches in the mentioned fields have been already introduced, however, they are in a preliminary stage and often are tailored to a specific problem, and do not allow the users to perform "complex" reasoning on the data. In order to make the utilisation of this paradigm computationally feasible and reliable in contexts where the reasoning methods have to handle a huge amount of data that "expires" soon, i.e., become soon useless, and quickly react to them, new solutions have to be introduced.

In this Thesis, we illustrate how *Logic Programming* can play a role in such challenging scenarios. We both describe the general approaches taken, and how these solutions have been used in several application contexts, some of which were milestones of large-scale international projects. We found that combining different reasoning methods and technologies is one of the crucial methodologies to adopt in order to effectively tackle and solve these challenges. Moreover, we identify the methodological gaps that are not yet closed, and prevent the large adoption of logic-based programming techniques.