Development and Application of Logical Actors Mathematical Apparatus for Logic Programming of Web Agents*

Alexei A. Morozov

Institute of Radio Engineering and Electronics RAS
Mokhovaya 11, Moscow 125009, Russia
AlexeiMorozov@netscape.net, morozov@mail.cplire.ru

One of the most interesting and promising approaches to programming Internet agents is logic programming of agents. This approach has good prospects, because the ideology and principles of logic programming are very convenient for searching, recognition, and analysis of unstructured, poorly structured, and hypertext information. Many ideas and methods of logic programming of Internet agents based on various modifications of Prolog and non-classical logic (linear, modal, etc.) were developed during the recent decade. Nevertheless, there has been no mathematical apparatus providing sound and complete operation of logic programs in the dynamic Internet environment (i.e., under conditions of permanent update and revision of information). To solve this problem, we have created a mathematical apparatus based on the principle of repeated proving of sub-goals (so-called logical actors).

Our mathematical apparatus for logic programming of Internet agents includes:

1. A model of intelligent agents that operate in a dynamical environment;
2. A classical declarative (model-theoretic) semantics of agents;
3. Control strategies for executing logic programs (Internet agents) that are sound and (under some conditions) complete with respect to the model-theoretic semantics of these agents.

Within the framework of our model of intelligent agents, an Internet agent (a group of Internet agents) is a logic program controlled by a special strategy. The control strategy is a modification of standard control strategy of Prolog, enhanced by so-called repeated proving of sub-goals.

The idea of repeated proving consists in dividing the program into separate sub-goals (called logical actors) [2, 3] that have the following properties:

1. Common variables are the single channel of data exchange between the actors.
2. Proving of separate actors can be fulfilled independently in arbitrary order.
3. One can defeat the results of proving of any actor while keeping all other sub-goals of the program.

* This work was supported by the Russian Foundation for Basic Research, project no. 03-01-00256.

After cancelling the results of proving an actor, its proving can be repeated. Thus, one can implement a modification of reasoning; the logical inference can be partially modified. This makes it possible to eliminate the contradictions between the results of the logical reasoning and new information received from the outside.

The most complicated and interesting problem to be solved for implementing the idea of logical actors and repeated proving is the development of control strategies supporting repeated proving that are sound and (if possible) complete. We have developed several control strategies supporting repeated proving.

One of the first control strategies was created for the execution of sequential logic programs with logical actors [2]. However, further experiments on visual logic programming have shown that it is expedient to develop concurrent control strategies as well.

To the present day, we have expanded our computing model by introducing concurrent processes and asynchronous messages. Our computing model is based on two kinds of messages in contrast to the standard OOP model. There are so-called flow and direct messages in our computing model [3]. The composition of messages of these two kinds helps us to describe the complex behaviour of agents without means of synchronisation of concurrent processes.

The main advantage of our computing model is that it provides a classical declarative (model-theoretic) semantics of concurrent logic programs. The logic programs are sound and (under some special conditions) complete with respect to their model-theoretic semantics. The completeness of logic programs means that the programming language guarantees that a program will found all solutions of a problem.

We have created an object-oriented logic language Actor Prolog on the basis of our mathematical apparatus (the definition of the language, including all new means, is available at our Web Site [1]). We have also introduced some special means that support programming of Internet agents in recent versions of the language. There are predefined classes implementing the HTTP and FTP protocols, some means for visual programming based on translation of Structured Analysis and Design Technique (SADT) diagrams into Actor Prolog [3], and syntactical features supporting component-oriented programming. Now, we have a working version of Actor Prolog. It supports the development of agents that automate of retrieval and analysing information on the Internet.

I am grateful to Prof. Yuri V. Obukhov, who is a co-author of the project.

References

1. Actor Prolog Web Site. http://www.cplire.ru/Lab144.
2. A.A. Morozov. Actor Prolog: an object-oriented language with the classical declarative semantics. In K. Sagonas and P. Tarau, editors, *Proc. of the IDL'99 Int. Workshop*, pages 39–53, Paris, France, September 1999.
3. A.A. Morozov and Yu.V. Obukhov. An approach to logic programming of intelligent agents for searching and recognizing information on the Internet. *Pattern Recognition and Image Analysis*, 11(3):570–582, 2001.