SADIO Electronic Journal of Informatics and Operations Research

http://www.dc.uba.ar/sadio/ejs

vol. 8, no. 1, pp. 1-11 (2008)

An Immune-based Approach to Student Diagnosis

Carine G. Webber[1] João Luís Tavares da Silva[1]

[1] Grupo de Inteligência Artificial e Sistemas Multiagentes
Departamento de Informática
Centro de Ciências Exatas e Tecnologia
Universidade de Caxias do Sul
Rua Francisco Getúlio Vargas, 1130
95020-972 Caxias do Sul, RS, Brasil
e-mail: cgwebber@ucs.br
No. tel. 55-54-32182100

Abstract

Biologically-inspired approaches constitute innovative problem solving techniques that have been applied to several domains including monitoring, detection and diagnosis. The human immune system (HIS) has especially motivated the development of new approaches to deal with problems where complexity and distribution are crucial constraints. Work in this paper reflects how characteristics from the HIS can be applied to conceive diagnosis systems. An application was implemented to student diagnosis to be integrated to Modal, an educational environment to the learning of basic programming skills.

Keywords: : Immune Engineering, Multiagent Systems, Student Modeling

1 Introduction

The human immune system is a complex of cells and organs able to carry out tasks such as pattern recognition, learning, memorization, generation of diversity, noise tolerance, distributed detention and optimization. Immune engineering has been studied as a new paradigm for solving non-deterministic problems, strongly founded by bottom-up methodologies.

Artificial immune systems have been applied in several domains such as data classification, pattern recognition, and detection [De Castro, 2000]. In our research, we consider the diagnosis as a kind of pattern recognition problem. Originally, a diagnosis task is the activity of observing a process (industrial, biological or sociological process), through sets of variables describing it, in order to characterize its state and ascribe a meaning to it (normal, abnormal or faulty state). Diagnosis, as a process, is started by the detection of changes in values expected for observable variables (constituting correct or incorrect patterns). Next, hypothesis must be generated to explain about the origin of such abnormal behavior.

Classical approaches to model diagnosis problems are: model-based diagnosis, abductive diagnosis and constraint-based diagnosis ([Console, 1989], [Reiter, 1987]). They are based on sets of rules, which describe system's normal and/or abnormal functioning. Recently, distributed approaches have appeared to deal with systems where components to be monitored are situated in different locations (industrial environment, computer networks, distant sensors and effectors). In this case, software agents can monitor components spatially distributed, interact to each other, share findings, and construct a complete diagnosis through cooperation mechanisms ([Frohlich, 1997], [Ross, 2003]).

Since previous works [Webber, 2002] we have been researching the possibility of taking a diagnosis as the result of the interactive activity of agents (from a micro level), result that is interpreted by specialized agents (macro level). In this sense we have implemented a few operational multiagent diagnosis systems alternating agents' behaviors from reactive (coalition formation based diagnosis) to cognitive approaches (ontology based coordination diagnosis). Following this approach, we have implemented an immune-based diagnosis system (named Artificial Immune System) whose principles are described in this article.

This paper is organized as follows. Next section briefly describes the HIS, whose characteristics have inspired this work. Section 3 focuses on the main direction of the research on artificial immune systems. Section 4 presents the domain of diagnosis and its application to the educational context. Section 5 illustrates the four steps of the immune-based diagnosis process proposed by this article. Sections 6 and 7 present Modal learning environment and the implementation under test. To conclude, last section discusses some relevant points for future research.

2 Human Immune System

The physiological function of the human immune system (HIS) is the defense against infectious microorganisms [Janeway, 2000]. It is known that the presence of an antigen can induce the body to produce specific antibodies, which generates the so-called immune adaptive response.

At the heart of the immune system is the ability to distinguish between self and non-self. Every body cell carries distinctive molecules that identify it as self. An antigen announces its foreignness by means of intricate and characteristic shapes called epitopes, which protrude from its surface. Most antigens, even the simplest microbes, carry several different kinds of epitopes on their surface; some may carry several hundreds.

However, some epitopes will be more effective than others at stimulating an immune response. The HIS is a complex of cells and organs able to carry out tasks such as pattern recognition, learning, memorization, generation of diversity, noise tolerance, distributed detention and optimization. It is composed by organs (bone marrow, thymus, spleen, and lymph nodes) and cells (granulocytes, macrophages, dendritic cells, B and T lymphocytes cells).

Lymphocytes are of two main types: B-lymphocytes (or B-cells) and T-lymphocytes (or T-cells). B-cells have the main function of producing antibodies in response to the presence of foreign proteins of bacteria, viruses, and tumor cells. Antibodies are specialized proteins that recognize and bind to one particular protein.

Antibody production and binding to a foreign substance or antigen is a mean of signaling other cells to engulf, kill or remove that substance from the body. T-cells are divided into two main classes. The first class includes the cytotoxic T-cells, which are important in directly killing certain tumor cells, viral-infected cells and sometimes parasites. The second class comprehends the T-helper cells, which main function is to increase the potential of the immune responses by the secretion of specialized factors that activate other white blood cells to fight off infection. Lymphocytes do not exhibit any functional activity until presented to an antigen, which is necessary to its proliferation and specific action.

Lymphocytes circulate continuously from the bloodstream to the lymphatic organs. When pathogenic microorganisms are captured in the lymphoid tissue, lymphocytes that recognize them are restrained there in order to proliferate and differentiate in effector cells, capable to fight off the infection. For the sake of brevity this is a short section, however we recommend [Janeway, 2000] for the study on HIS.

3 Artificial Immune System

The HIS has inspired the conception of 4 main algorithmic approaches namely Negative Selection [Forrest, 1984], Danger Models ([Matzinger, 2002], [Aickelin, 2002]), Clonal Selection ([Weinand, 1990], [De Castro, 1999]), and Immune Network Models ([Farmer, 1986], [Fukuda, 1993]). Garrett (see [Garrett, 2005]) presents an important study on the usefulness of these methods comparing to existing ones (for instance, genetic algorithms, neural networks, hidden Markov models) considering their distinctiveness and effectiveness to solve problems. Although AIS is a young domain of research, the author argues that they are promising methods (to be applied alone or in conjunction to other methods) producing in most cases effective results.

This article is concerned with artificial negative selection (ANS). Natural negative selection (NNS) is the ability that mature B and T lymphocytes have to distinguish self (body's cells) from non-self (antigens). NNS provides tolerance to self cells (avoiding self-immune diseases) and induces the appropriate immune response to antigens.

ANS has been commonly applied to detection and diagnosis tasks. Forrest was the first to propose and apply negative selection algorithms to abnormal detection. A negative selection algorithm is an abstract model, which defines “self” by building normal behavior patterns of a monitored system. Detectors are intended to detect when elements of the self-set have changed from an established norm.

The original algorithm of negative selection works on a set of self-binary strings, then it generates new strings, and calculates their similarity to self strings. If a new string is similar to self, then it is discharged; otherwise it is added to the detector set. Once the detector set is built, it helps to discriminate non-self elements. Dasgupta (see [Dasgupta, 2006]) describes different negative selection algorithms considering particular representation schemes, matching rules and detector generation processes.

For the purpose of this work, we have conceived a diagnosis multiagent system whose behavior is based on how the HIS distinguishes self from non-self. We have been inspired by ANS principles, although we have not worked with strings to represent self and non-self. According to our model of self/non-self discrimination, agents encapsulate rules of normal and abnormal behaviors and together they verify inconsistencies. System’s domain of application and implementation are presented in the two next sections.

4 Domain of Application

The Artificial Immune System (AIS) was conceived as a multiagent system applied to student diagnosis in the context of a learning environment. An important aspect of learning environments is the ability of taking into

account students' knowledge in order to generate new learning situations or to intervene during problem solving activities. The student diagnosis is a way of taking into account students' knowledge.

The AIS is an abstraction of the HIS, whose main characteristics are the interaction among its components, the reactive behavior face to intruders, the complex environment (the body), the ability of self-adapting and self-organizing, pattern recognition, memory and the capacity of operating simultaneously in different portions of the environment.

We have integrated the AIS to a learning environment called Modal. Modal was conceived to help students learn the basic skills of programming using a structured language similar to Pascal. Programming is a complex skill to learn and, as already known, compiler tools are not originally conceived to be used in educational contexts because learners easily develop "generate and test" problem solving methods. As a consequence, students may hold misconceptions about the use of language statements and programming logic. Modal intends to fulfill such learning requirements.

As a multiagent system, Modal was implemented using the generic multiagent platform FIPA-OS and the PMA3 platform. PMA3 offers the common infrastructure for conceiving multiagent learning environments in any domain, comprehending agents for pedagogical and communication services [Webber, 2006]. Among its components, Modal has lexical, syntax and semantic analyzers of the structured language that students must use to interact with the environment. Lexical and syntax analyzers are each one a single agent. The semantic analyzer was decomposed in several small agents, each one responsible for checking whether a rule holds. The agent If-Agent, for instance, is in charge of the correct use of the if-then-else statement of the language. Besides this one, other agents compose the semantic analyzer (for instance, declare, write, attrib, switch, read, uniquevar, for, while, and repeat agents).

Modal agents were implemented for a previous approach of the student diagnosis based on the coordination of agents and the semantically distributed diagnosis model [Webber, 2002]. Such approach appeared to be quite complex. For this reason, we have decided to explore the immune-based approach to model agents` behavior and diagnosis convergence.

Figure 1 illustrates the two levels given by Modal agents and AIS, as well as its interactions. Four classes of agents compose the AIS architecture: a macrophage, T-lymphocytes, B-lymphocytes, and a diagnosis Blymphocyte. They were implemented over Modal agents based on their functionalities.

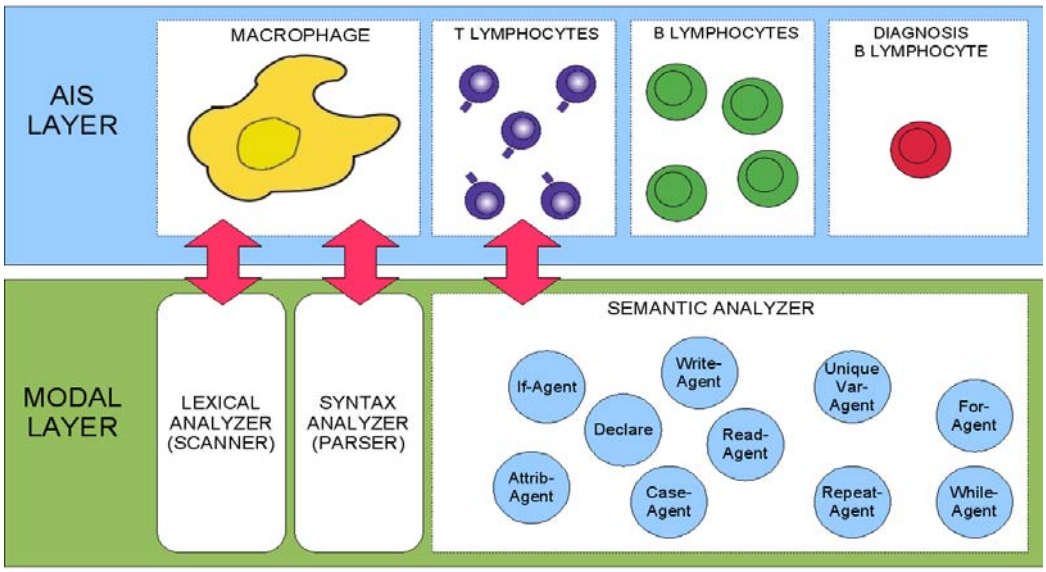

Fig. 1. Modal and Artificial Immune System's Agents

As showed, lexical and syntax analyzers are part of macrophage agent. Semantic agents are part of Tlymphocytes. The student's solution is the antigen.

5 Artificial Immune System Description

The immune-based diagnosis occurs on four steps namely phagocytosis, adaptive immune response, sensitization of B-lymphocytes, and B-lymphocyte diagnosis. Each step is briefly described next.

Step 1 - Phagocytosis

Phagocytosis is the process where the macrophage engulfs the infectious microorganisms (antigen). In the context of AIS, the macrophage agent recognizes through its sensors an antigen (a sequence of programming statements), and then phagocytes it.

Once the process of phagocytosis is started, the antigen is broken into pieces (similar to a process of tokenization). Each token is called a peptide. A peptide corresponds to one statement either to input and output data (read and write), to control loops of execution (repeat, while, and for), to test a condition (if-then-else and switch), set values to variables, constants, or to compute arithmetic/logic operations. Figure 2 illustrates the phagocytosis process, where a student's solution is the input to macrophage agent. Small squares represent the peptides (tokens like "read", "+", "if", "x", and the assignment operator "<-"), which originally concentrate on the surface of the macrophage.

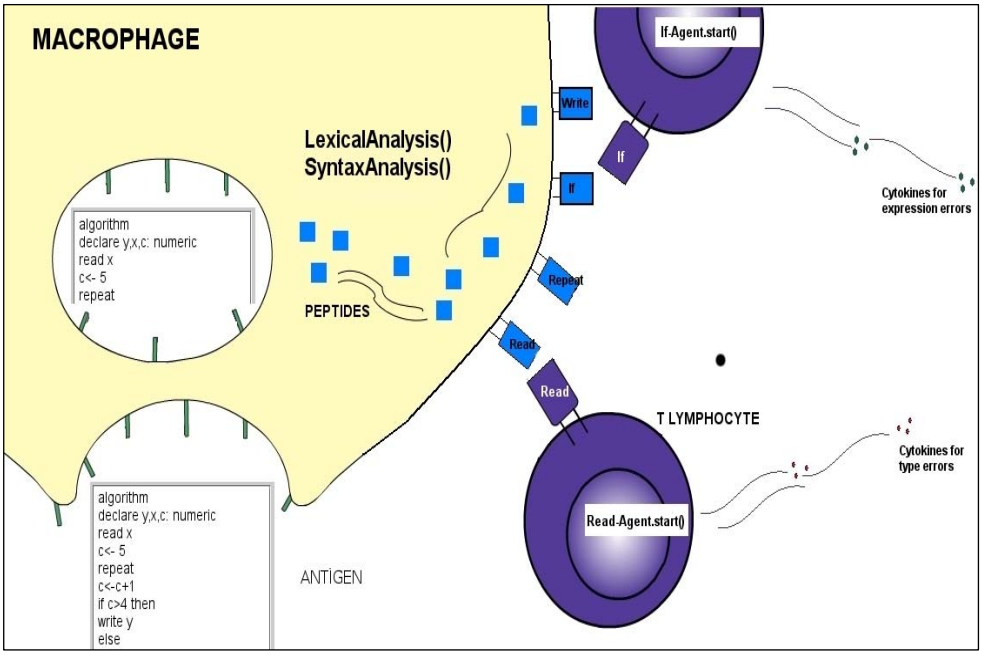

Fig. 2. Phagocytosis process

At this step of the process, a new data structure is needed to store the peptides on the surface of the macrophage agent. Peptides must be located at the surface of the macrophage in order to be readable by Tlymphocytes (as showed at figure 2). This first step corresponds to the innate immune response observed on HIS, which provides immediate defense against antigens.

Step 2 - Adaptive immune response

When peptides become available for T-lymphocytes, the adaptive immune response initiates. This response brings to the HIS the ability to recognize certain pathogens. Each T-lymphocyte agent searches for one instance of a peptide having affinity with its receiver. A receiver recognizes peptides (same tokens like "read", "+", "if", "x", "<-"). Once a T-lymphocyte agent recognizes a peptide, it activates a particular semantic agent (illustrated on figure 1). Semantic agents realize the distributed semantic analysis of the student´s program, where each agent seeks for abnormal use of one statement.

If one semantic agent recognizes the peptide carries a semantic error, the T-lymphocyte agent discharges a cytokine message in the environment (figure 2). Cytokines are originally proteins that affect the behavior of certain sensitive human body cells. In the AIS, cytokines correspond to messages exchanged among agents through a blackboard architecture. We defined four classes of cytokines (types of errors):

-Expression error: errors using regular language statements (if-the-else, repeat, write, and so on);
-Type error: mismatch between variables and values;
-Unicity error: errors declaring variables;
-Declaration error: signals an incompatibility between the declaration type of a variable and its use.

A cytokine message is composed by the name of the agent secreting it, its class (expression, type, unicity or declaration), and a description. Examples of cytokine messages are as follows:

```
(LymphoTAgent1 (expression_error) ("invalid expression" while_statement
line_15))
(LymphoTAgent2 (declaration_error) ("incompatible value attribution"
value_attribution line_21))
```

Once T-lymphocyte agents stop secreting cytokines, they become inactive.

Step 3 - Sensitization of B-lymphocyte agents

We have implemented four B-lymphocyte agents. Each one is sensitive to one of the cytokine's classes of errors (expression, type, unicity or declaration error). As soon as cytokines are secreted, B-lymphocyte agents become active and monitor the environment. When a B-lymphocyte recognizes the particular cytokine it is sensitive to, it stores cytosine's information. As a result, each agent forms groups of errors related to the same category. Once no more cytokines are found, each B-lymphocyte secretes a diagnosis cytokine message, destined to the special diagnosis B-lymphocyte. At the end of this step, B-lymphocytes become inactive.

Step 4 - Diagnosis B-lymphocyte agent

The last step concludes the diagnosis task. It is supported by a diagnosis B-lymphocyte agent, which is sensitive to cytokines of diagnosis previously secreted. Diagnosis B-lymphocyte agent captures partial diagnosis messages, organizes and integrates them in order to build a final diagnosis. Final step is illustrated on figure 3.

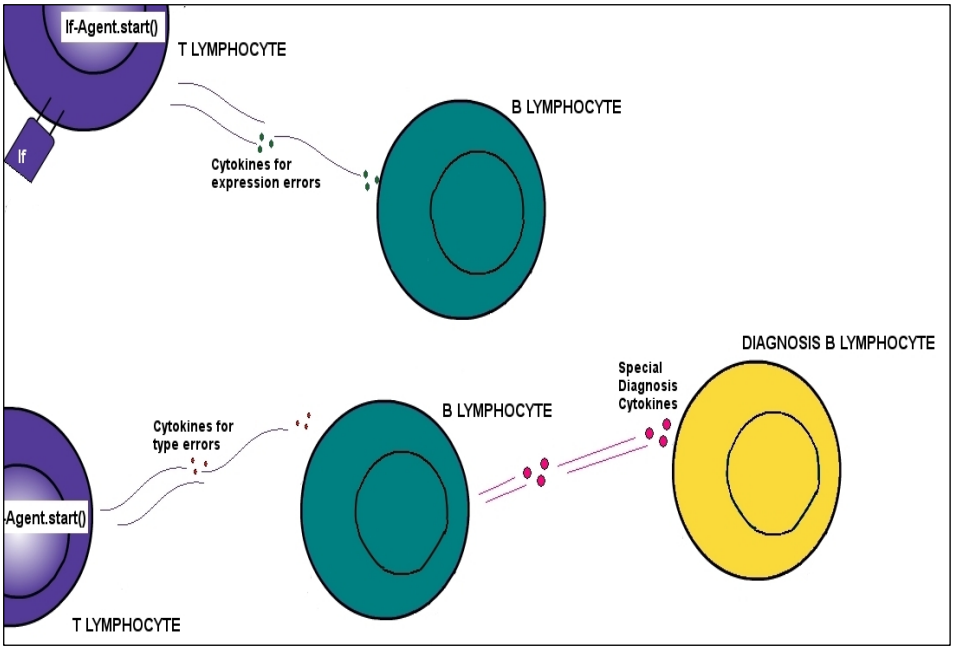

Fig. 3. Flow of cytokine messages

Final diagnosis comprehends syntactic and semantic errors recognized and grouped according previous classes of errors. Modal system retrieves and uses this information to customize learning activities, to improve interactions, and better evaluate student's learning. Syntactic errors can be pointed out to the student, to explain or recall the student about the correct use of statements. Semantic errors are interpreted as student misconceptions, following an educational theory, being treated accordingly to learning priorities and/or student's difficulties. Since such aspects go beyond the objective of this paper, we refer the reader to [Webber, 2002].

6 Modal Learning Environment

Modal is a JSP (Java Server Pages) application composed of several agents. These agents can propose exercises, help in editing programs, compile solutions, and provide asynchronous and synchronous communication channels among learners. Figure 4 illustrates student's interface presenting an interaction between the student and a teacher during problem solving activities. According to resources available on both workstations, student and teacher are able to see and communicate to each other.

Modal was conceived to help students learn the basic skills of programming using a structured language similar to Pascal. Each student has access to a library of exercises and to a private folder. During problem solving activities students can also check whether their solutions are correct. Students are asked to construct programs, to test them, and correct them until they run properly.

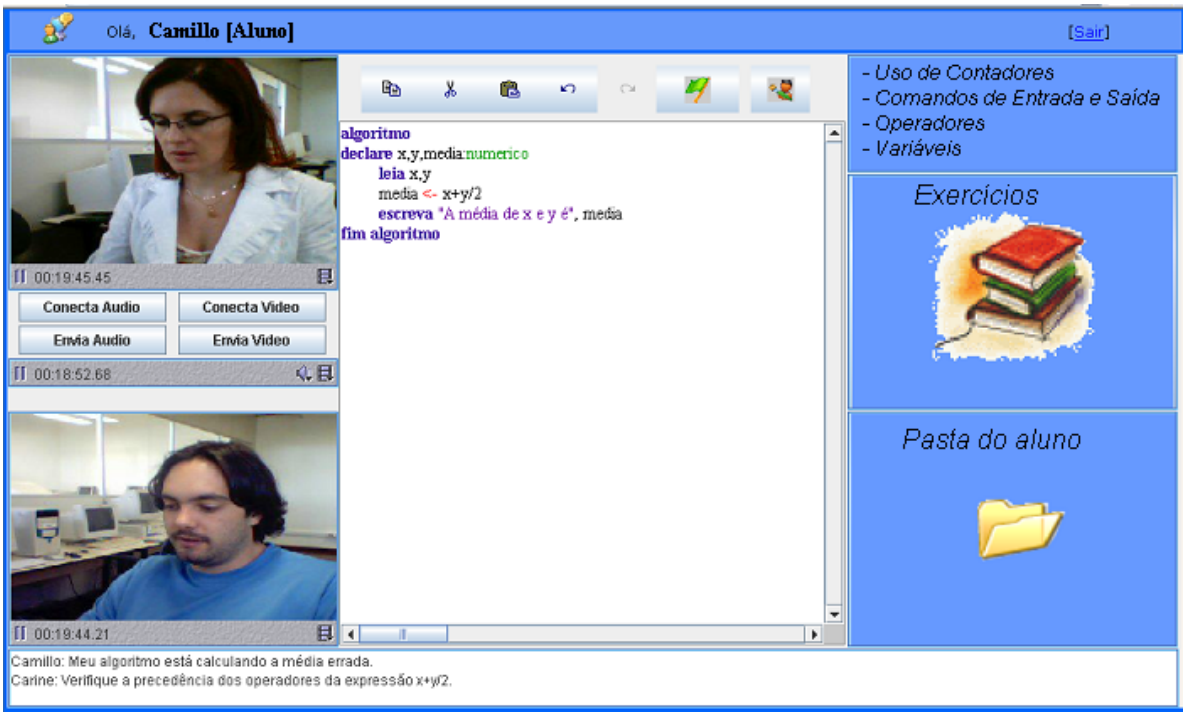

Fig. 4. Modal student's interface

Modal informs students as a regular compiler about their syntax errors, recording information about student's difficulties. Since 2003 we have been collecting novice-programming solutions in order to observe which are the common errors we find. This research is important as long it sheds light on misconceptions students may hold, which allows developing better adapted learning environments.

We have observed during learning activities that there are two main situations that occur when students face difficulties solving problems. Some students try to solve a problem and give up without seeking for alternative paths to solve it, lacking of confidence. On the other hand, some students keep on trying different paths, repeating them and going around in circles. Usually such behaviors point out that students do not fully understand the semantics of the programming language constructs. Some studies have shown that students lacking a mental model of the language (statements, syntax, etc) were unable to decompose a problem in order to solve it. Our research has confirmed these assumptions, recognizing that one main cause of novice errors concerns the difficulty of associating a problem to a set of primitives that solve it correctly. Also, during learning activities, novices tend to develop misconceptions about the use of such primitives. A typical misconception students hold concerns assignment, where A=B is interpreted in a way that B is linked to A and any future changes to A affect on B. This is maybe due to the fact that in Mathematics assignment and logic operators are represented by the same symbol (x=4 ; 9=13-4). Input statement (read) causes as well some trouble in understanding the hidden kind of assignment. Students often do not understand how a read instruction interrupts execution until it receives a value from the user.

At this point, we find that our approach of distributed diagnosis is particularly suitable since it decomposes student's program in order to better analyze the different aspects having influence on students reasoning (syntax use of primitives, semantic meaning of words, senses to components of the program, among other more subtle aspects of writing a program). First experiments reported in this article have focused on learning logical and mathematical expressions. We have chosen this topic of knowledge because constructing correct expressions is a basic but primordial skill novices must develop to pursue learning of complex structures. Expressions control the execution of a program (when used to test conditions and to interrupt loops), but also they involve variables, constants, and operators. Such diversity allows the system to check a great variety of problem situations and misconceptions students may hold.

7 Implementation

We have been testing AIS diagnosis with real students' solutions using a special interface we have implemented to observe agents' behaviors during diagnosis task (figure 5). On the left side of the window, student's program is typed. On the right side (on top), messages exchanged through the blackboard can be checked. On the right bottom, partial and final diagnoses are presented.

A simulated run of the system consists of 20 agents able to recognize self (correct use of expressions) and non-self (incorrect use of expressions). As previously described, agents encapsulate a few self or non-self rules to discriminate correct constructions of expressions form erroneous ones. Programming language is a Portuguese Pascal-like language. So far students are requested to use distinct symbols for the logic operator ("=") and for the assignment operator ("<-"). We report here some preliminary results based on our investigation of different student's programs.

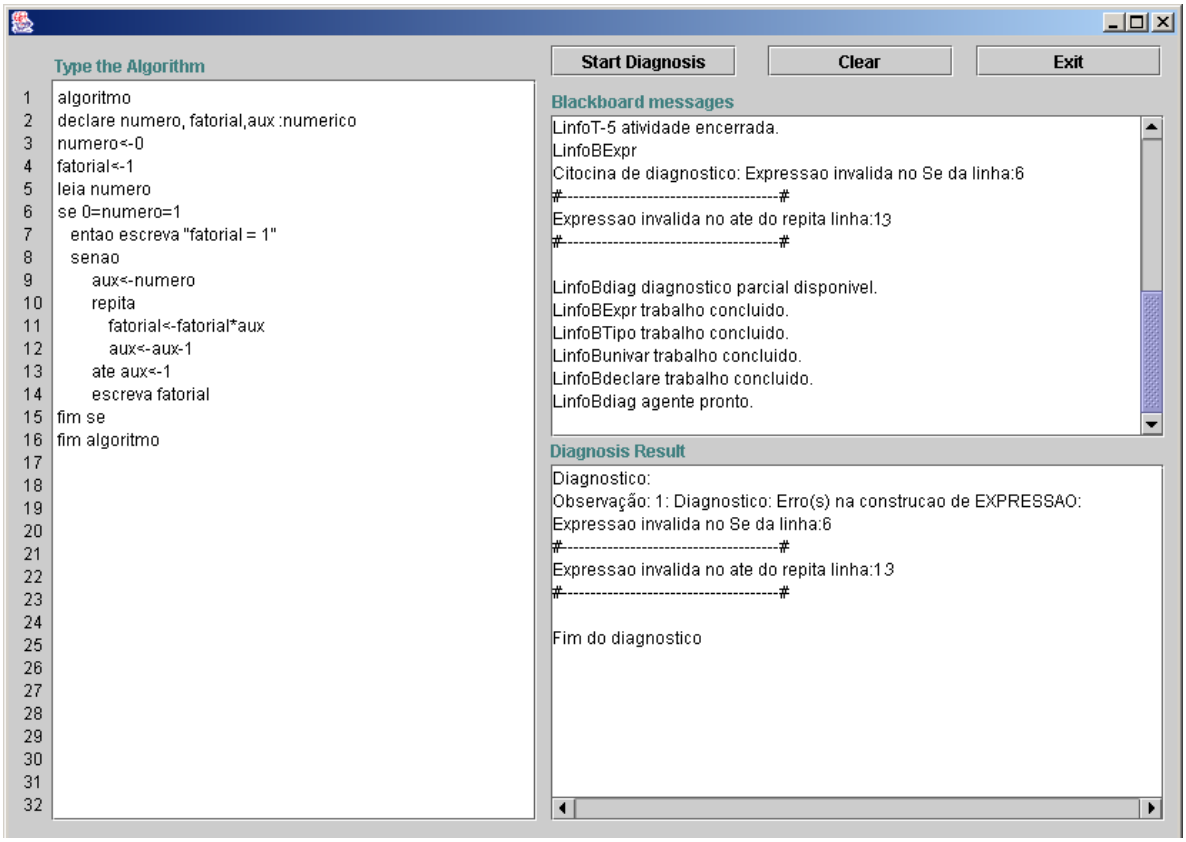

Fig. 5. AIS test interface

We have requested new learners to write a program which reads a number, calculates its factorial number if it has one, and outputs it. This is a typical problem new students are faced with in the beginning of our programming course. We have observed common errors students make when ascribing values to variables, controlling loops, and calculating the factorial itself. An example of system evaluation is shown on table 1.

Student´s solution is presented on the left side of the table (a). Two gray lines (lines 6 and 13) illustrate two common wrong expressions. On line 6, the student concatenated two comparisons that could mean "numero=0 or numero=1", but also "numero=0 and numero=1". On line 13, the student misemployed assignment operator to define a loop stop condition (aux<-1). On (b), an excerpt from the blackboard of cytokines secreted by T-cells describes messages that allowed the inference of final diagnosis on (c).

At this moment of this project, we expect agents to recognize that errors related to if-then and repeat statements are actually caused by wrong constructions of both expressions. This observation leads to a conclusion that the system must interpret as a misconception on “test-condition expressions”, not meaning the student does not understand if-then or repeat statements.

Table 1 – Diagnosis summary

(a) Original student´s solution	**(b) Translated Cytokine messages**
(1) algoritmo (2) declare numero, fatorial, aux :numerico (3) numero<-0 (4) fatorial<-1 (5) leia numero (6) se 0=numero=1 (7) entao escreva "fatorial = 1" (8) senao (9) aux<-numero (10) repita (11) fatorial<-fatorial*aux (12) aux<-aux-1 (13) ate aux<-1 (14) escreva fatorial (15) fim se (16) fim algoritmo	[...] Lymphocyte T-0: cytokine type 'expression' semantic agent: AgenteSe (if statement), error: invalid expression on (IF) line 6 Lymphocyte T-1: cytokine type 'expression' semantic agent: TipoRepita (repeat statement) error: invalid expression on repeat line 13 [...]
	(c) Translated Diagnosis
	Finding1: Error(s) in EXPRESSION: Invalid Expression on IF (SE) line 6: Invalid Expression on repeat (ATE) line 13:

So far, we have been interested on modeling and implementing an immune-based approach to student diagnosis and on validating it using students solutions. Next step will consist on extending our testbed to other aspects of learning programming besides logical and mathematical expressions. We plan to integrate the AIS to Modal interface as long as more agents are developed and tested. Improvements on performance will be expected as well.

8 Conclusion

We have implemented an approach to diagnosis where sets of agents having complementary functionalities are assigned to solve a diagnosis problem in a bottom-up methodology. Diagnosis functionality is not reducible to the functionality of one specific agent, but it *emerges* from the interaction of several agents.

The hierarchical organization of agents is an important aspect of the AIS implemented, since there is a clear dependency among expected agents’ behaviors. This becomes a crucial aspect, since the number of agents in the society tends to grow fast. Related work on this subject is been considered in the present.

So far, the outcome of this work demonstrates that immune engineering is promising to student diagnosis. The research project continues on implementing new agents to increase the classes of errors to be treated. We have been searching as well for relevant criteria to compare the immune-based approach to diagnosis with other operational diagnosis applications.

Acknowledgement

This work was supported by University of Caxias do Sul and FAPERGS.

References

Aickelin, U., Cayser, S. The danger theory and its application to ais. In Proceedings of the First International Conference on Artificial Immune Systems (ICARIS-2002), pp.141-148, 2002.

Console, L., Theseider Dupré, D., Torasso, P. “A theory of diagnosis for incomplete causal models”, *Proceedings of the 10th IJCAI*, USA, 1989, pp.1311-1317.

Dasgupta, D. Advances in Artificial Immune Systems. IEEE Computational Intelligence Magazine, November, 2006. pp.40-49.

De Castro, L. N. & Von Zuben, F. J., "Artificial Immune Systems: Part I – Basic Theory and Application", Tech.Report, Unicamp 01/1999, p.65.

De Castro, L. N. & Von Zuben, F. J., "Artificial Immune Systems: Part II – A Survey of Applications", Tech.Report (available on internet), Unicamp 02/2000, p.65.

Farmer, J., Packard, N., Perelson, A. The immune system, adaptation and machine learning. Physica D, 22, pp.187-204, 1986.

Forrest, S., Perelson, A.S., Allen, L., Cherukuri, R. “Self-Nonself Discrimination in a Computer”, In Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy, Los Alamitos, CA: IEEE Computer Society Press, 1994.

Frohlich, P. M'ora, I.A., Nejdl, W., Schroeder, M. “Diagnostic agents for distributed systems”, *Proceedings of ModelAge97*, Sienna, Italy, 1997.

Fukuda, T., Mori, K., Tsukiyama, M. Immune networks using genetic algorithm for adaptive production scheduling. In : Proceeding of the 15th IFAC World Congress, volume 3, pp.57-60, 1993.

Garrett, S.M. How Do We Evaluate Artificial Immune Systems? Evolutionary Computation, 13(2), pp.145-178, 2005.

Janeway, C., Travers, P., Capra, J.D., Walport, M.J., *Immunobiology: The Immune System in Health and Disease*, Garland Pub, 2000, pp.635.

Matzinger, P. The Danger Model: A renewed sense of self. Science, 296 (5566), pp.301-305, 2002. Reiter, R, “A theory of diagnosis from first principles”, *Artificial intelligence* , 32 (1), 1987, pp.57-96.

Ross, N., Teije, A., Witteveen, C, “A Protocol for Multi-Agent Diagnosis with Spatially Distributed Knowledge”, Austrália, *ACM Press*, 2003, pp.655-661.

Webber, C., Pesty, S. “Emergent Diagnosis via Coalition Formation”, *LNCS*, S.Verlag, n.2527, 2002, pp.755-764.

Webber, C., Prado Lima, M.F.W., Casa, M.C., Ribeiro, A.M., “Adding Security to a Multiagent Learning Platform”, *The 1st Int. Conf. on Availability, Reliability and Security*, IEEE Computer Society, Austria, 2006, pp. 887-894.

Weinand, R.G. Somatic mutation, affinity maturation and antibody repertoire: A computer model. Journal of the Theoretical Biology, 143, pp.343-382, 1990.