The Biological Basis of the Immune System as a Model for Intelligent Agents

Roger L. King[1], Aric B. Lambert[1], Samuel H. Russ[1], and Donna S. Reese[1]

[1]MSU/NSF Engineering Research Center for Computational Field Simulation,
Box 9627, Mississippi State, MS 39762-9627, U.S.A.
Rking@erc.mstate.edu

Abstract. This paper describes the human immune system and its functionalities from a computational viewpoint. The objective of this paper is to provide the biological basis for an artificial immune system. This paper will also serve to illustrate how a biological system can be studied and how inferences can be drawn from its operation that can be exploited in intelligent agents. Functionalities of the biological immune system (e.g., content addressable memory, adaptation, etc.) are identified for use in intelligent agents. Specifically, in this paper, an intelligent agent will be described for task allocation in a heterogeneous computing environment. This research is not intended to develop an explicit model of the human immune system, but to exploit some of its functionalities in designing agent-based parallel and distributed control systems.

1 Introduction

Biological systems are serving as inspirations for a variety of computationally-based learning systems (e.g., artificial neural networks and genetic algorithms). This research uses as its biological inspiration the human immune system. From a computing standpoint, the immune system can be viewed as a parallel, distributed system that has the capability to control a complex system over time [1]. The primary objectives of this research are to gain an understanding of the recognition process, learning, and memory mechanisms of the immune system and how to make use of these functionalities in the design of intelligent agents [2, 3].

The immune system is comprised of several different layers of defense -- physical, physiological, innate (non-adaptive), and adaptive. The adaptive defense mechanism is sometimes referred to as *acquired*. Any action of the immune system against a pathogen is known as an *immune response*. The most basic defense mechanism is the skin. It serves as a physical barrier to many pathogens. For pathogens that elude the skin barrier, there are physiological barriers. These are immune responses that provide environments non-conducive to pathogen survival

(e.g., an area of low pH). These static (physical and physiological) immune responses are of minor interest in this research.

The important difference between innate and adaptive responses is that an adaptive response is highly specific for a particular pathogen. This occurs because the innate system agent will not alter itself on repeated exposure to a given pathogen; however, the adaptive system agent improves its response to the pathogen with each encounter of the same pathogen. Also, this adapted cell will remain within the body and will be able to attack the pathogen if it re-enters the system in the future. Therefore, two key functionalities to exploit of the adaptive immune response system are its abilities for *adaptation* and *memory*. The following sections explore in more detail how adaptation and memory occur as a result of the immune response process.

2 Immune Response

Innate and adaptive immune responses are produced primarily by leukocytes. There are several different types of leukocytes, but the most important for our consideration are phagocytes and lymphocytes. The phagocytes are the first lines of defense for the innate immune system. These cells use primitive, non-specific recognition systems that allow them to bind to a variety of organisms, engulf them, and then internalize and destroy them. An interesting strategic maneuver on the part of the immune system is that it positions phagocytes at sites where they are more likely to encounter the organisms that they are most suitable to control. In other words, the immune system does a judicious job in controlling its limited resources. Lymphocytes initiate all adaptive immune system responses since they specifically recognize individual pathogens. The two main categories of lymphocytes are B lymphocytes (B-cells) and T lymphocytes (T-cells). The B-cells develop in the bone marrow and the T-cells develop in the thymus.

Immune response is a two step process - *recognition* followed by *resolution*. It is also proposed that immune system based intelligent agents for task allocation utilize a two step process - *recognition* of a specific hardware and/or software instance followed by an *allocation response* (i.e. a resolution) that will better utilize the computing environment's resources to perform on-going and planned tasks. Two types of intelligent agents are proposed to accomplish the recognition process - H-cells specialized for hardware and S-cells specialized for software. Details of their functionalities and use are described later.

Pathogen recognition is the first step in the adaptive system's response to an organism. There are several mechanisms through which this can occur - the complement system, cytokines, and antibodies. Most applications of the immune system to computing system research have concentrated on the antibody recognition system [4-9]. Although this will play an important role in this research, it is believed that the cytokine system offers other desirable functionalities that should be exploited.

Cytokines are a class of soluble mediators released by T-helper cells (a specific type of T lymphocyte) to communicate with other cells in the innate and adaptive immune systems. For example, these T-helper cells work with the innate system's phagocytes or the adaptive system's B-cells. In the innate system case, phagocytes

take up antigens and present them to the T-helper cells to aid in antigen recognition. In return, if the T-cell recognizes it as an antigen, it will release cytokines to activate the phagocyte to destroy the internalized pathogen. T-cells also work with B-cells to help them divide, differentiate, and make antibody. The principal to recognize here is that although the agents are distributed, many of them communicate with each other to facilitate optimal control. The functionality exhibited by the cytokine system which we would like to exploit for our agents are the *division of responsibility among distributed agents for command, control, and communication*.

3 Recognition, Adaptation, and Antibodies

B-cells manufacture antibodies. However, each B-cell is monoclonal (i.e., it only manufactures one type of antibody) and so it is limited to recognizing antigens that are similar in structure to itself. Recognition of an antigen by an antibody takes place at a special region on the antibody molecule – the paratope. The paratope is the

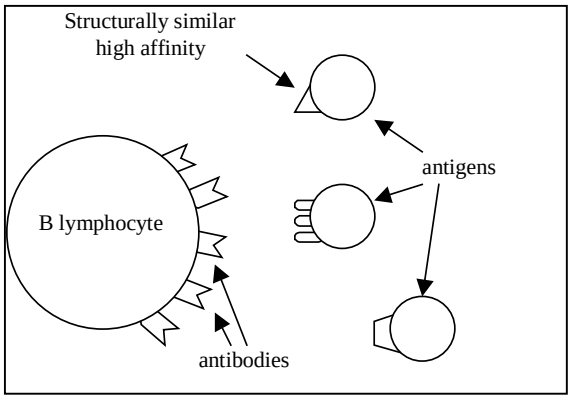

Fig. 1. Antibody – antigen recognition and binding.

genetically encoded surface receptor on an antibody molecule. The area on the antigen where a paratope can attach itself is known as the epitope. When a particular antigen is recognized or detected by a B-cell, it combats the antigen by reproducing in numbers (i.e., cloning) and by releasing its soluble antibodies to bind to the target antigen. It should be noted that an antibody produced by a B-cell is specific for an epitope and not the entire antigen molecule. In fact, several different antibodies from different B-cells may bind to a single antigen.

Antigen recognition or detection is accomplished via a process known as binding. The binding of an antibody to an antigen is a result of multiple non-covalent bonds. The analytical representation often used for this antibody-antigen bond is a complementary binary string (a visual representation would be a lock and key) [2]. For example, if a paratope on a B-cell was represented as 01100110, then the epitope on an antigen that would exactly fit this lock would be 10011001 and the binding energy would be at its maximum. Antibody affinity is a measure of the strength of the bond between the antibody's paratope and a single epitope. Therefore, it can be

surmised that structurally similar epitopes will also bind to this antibody, however, with lesser energies. This gives rise to the concept of an affinity threshold. If the threshold is exceeded, recognition has occurred and the B-cell is cloned. For an intelligent agent, if real values are used it may be necessary to develop a fuzzy metric relationship. However, if binary representations are used for resource loading, software preferences, etc., then Hamming distance or an XOR approach may be a good metric for threshold affinity. This is illustrated in Figure 2. In this example, if the threshold affinity was 10, then antibody 2 would have exceeded the threshold and cloned itself.

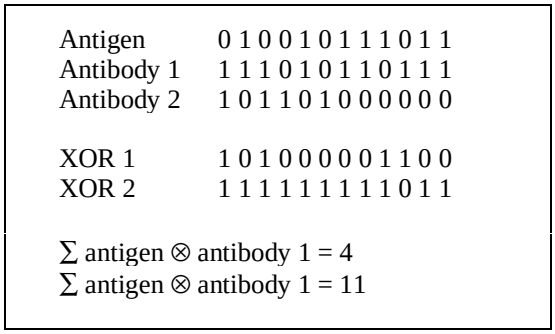

Fig. 2. Illustration of binding energy calculation.

It was just observed that when a B-cell exceeds its affinity threshold it clones itself. An important aspect of this cloning process is that it does not produce exact clones. The offspring produced by the cloning process are mutated. This variation from child to parent is important in adapting B-cells to better fight antigens. The rationale behind this inexact cloning can be explained as follows. Because the parent probably had a less than maximum affinity when it was stimulated, an exact copy of the B-cell would not improve the response of the immune system. However, if the parent's offspring are mutated, then there is a chance that the children will have a higher affinity for the antigen, and then, subsequent generations can improve their response. Therefore, the immune system exhibits the functionality of *evolutionary adaptation* (i.e., learning ala Darwin). Learning through adaptation is a functionality of intelligent agents for task allocation.

4 Immunological Memory

Memory is another important functionality existing within the human immune system. The above description of evolving B-cells to better fight a particular antigen is known as *the immune system's primary response*. This initial encounter with a new antigen involves a finite amount of time for the immune system to adapt its B-cells to best combat the intruder. And, when fighting a pathogen, time is of the essence. To improve response time, the immune system has developed a way to maintain a memory of this encounter so it will be able to respond faster in the future (i.e., have

antibodies already available). Then, when the body is re-infected with the same antigen, the secondary response of the immune system, based on *immunological memory*, is very specific and rapid. This ability of the immune system to remember instances of previously encountered activities and the learned response is another functionality of immune systems to be exploited in intelligent agents. In fact, immunological memory is the functionality of the human immune system that vaccinations exploit. With this analogy in mind, it very quickly follows that *a priori* information about a heterogeneous computing environment can be initially known via vaccinations with appropriate H-cells and S-cells. The inoculation could provide H-cells containing information about the system's initial hardware resources and the S-cells could contain explicitly declared information about the code (e.g., what tasks are read-write intensive or how much processor time will the task need).

Immunological memory may best be described as a *content addressable memory* (CAM). The CAM functions fundamentally different from the classical location addressing memory of most computer architectures (i.e., von Neumann architecture). In the immune system an antigen may be addressed for recognition purposes simultaneously by many B-cells (i.e., in parallel). When the affinity threshold of a specific B-cell is exceeded by either an exact or approximate match, then a specific response is elicited from memory (clone, release antibodies, destroy). The key functionality of a CAM is that the content of the stimulus is important in eliciting a response specific to that stimulus. In-other-words, an instance of a previously encountered hardware configuration and/or software requirement will be recognized by the system and the learned response (task allocation) associated with this instance will be specifically and rapidly dispatched from memory. Therefore, the secondary response of the intelligent agents (H-cells and S-cells) should improve over their primary response due to the retained memory of their previous encounter.

Another important point about immunological memory is that it is *self-organizing*. It must have this capability because the immune system is limited in the number of cells it may have available to it at any one time. Since the number of cells is limited, the immune system replaces a percentage of its lymphocytes on a continuing basis. This replacement or self-organizing takes place during the learning phase of the cell. It was explained previously that immune system learning was evolutionary in the sense of Darwin (i.e., survival of the fittest). Therefore, if cells are not periodically matched with a pathogen, then they will die out to be replaced by a new cell that may better match existing or previously encountered pathogens in the body. Since the number of instances of hardware configuration and software architecture can be infinite, it will be important for the intelligent agents to have the capability of purging poorly performing H-cells and S-cells from the system. The measure of quality of performance used by the self-organizing functionality of the resource allocator may be based on a performance index, a time since last used, or some other set of metrics.

One of the primary functions of the body's immune system is to perform classification of organisms as either self or non-self (i.e., your own versus foreign). This capability is distributed among a variety of agents (adaptive and non-adaptive) whose function it is to continually monitor the body's environment for these undesirable intruders. If an organism is classified as self, then no actions are taken. However, if an entity is classified as non-self, the immune system's agents work autonomously and/or in concert to eliminate the organism.

5 H-cells and S-cells as Intelligent Agents for Task Allocation

H-cells and S-cells are envisioned as being designed as complementary intelligent agents. The S-cells have the role of defining the long term schedule of resources for execution of a program (i.e., a road map) and the H-cells provide near real-time control of the schedule (i.e., adjustments for the bumps and curves in the road). Where H-cells interrogate and monitor hardware resources during execution of a program for information, the S-cells interrogate the parallel program code for information prior to execution. This information is then used by the S-cells in planning how to schedule the code on the known distributed system hardware resources. The H-cells and S-cells are also distributed throughout the heterogeneous computing environment to focus on their assigned tasks, thus decentralizing the control process.

Initially, the S-cells may determine whether the code is data domain intensive or functional domain intensive. This determination can be made through either programmer supplied information or as a learned response to the program as it is executed. For example, in a data domain intensive code, the tasks to execute on all the hardware resources is functionally the same. However, the data may be different. In this case, knowledge about the nuances of the data (file size, precision requirements, etc.) will be important in assigning resources. In the case where the functionality of tasks differs (i.e., functional domain intensive), it is important to have knowledge about the resource requirements of tasks (e.g., memory requirements, execution time, etc.). A simulation is a good example of a functionally intensive domain. Tasks will vary among the resources with different constraints for communication, execution, etc. The S-cell has the responsibility of determining critical paths for the simulation and allocating its distributed resource base in a judicious manner.

In the case of the H-cells, they are initially inoculated with basic knowledge about system hardware resources and behavioral properties. They will then have the tasks of learning about new hardware resources connected into the environment, monitoring resource usage, and managing system resources and processes. Inputs to the H-cells may include memory availability, CPU usage, disk traffic, and other performance related information. For the prototype S-cells it is envisioned that the software programmer will embed information about the code into pre-defined header slots. For example, information about critical paths may be specified, along with which tasks are computationally or communicationally intensive. Then, when a program is launched, the S-cells continue to monitor the activity of resources that they have planned to utilize along the execution path. If one of these resources becomes unavailable, it will be the responsibility of the H-cells to find an appropriate available resource in the short term. However, the S-cells will then reassess the road map and determine what resources it had planned to use to complete the tasks that may have to be remapped. After making any necessary adjustments to the road map, the new routes will be passed to the master unit for any appropriate action.

At present, research is on-going to develop intelligent agents to perform task allocation for a heterogeneous computing environment [10]. Implementation of many of the functions of the human immune system described in this paper are being demonstrated in an H-cell implementation based on a hierarchical ART architecture [11]. Data for inoculation of the H-cells and for defining the content addressable

memory are being collected at both the system and task levels. At the system level, information is both static and dynamic. The static information includes: machine type, processor speed, operating system, internal memory available (both physical and virtual), page information, ticks per second, and number of processors. At the task level, all of the information is dynamic in nature. Task information includes: memory usage and a breakdown of a task's CPU usage into time for computing and communicating while executing an MPI task.

5 Implementation in the HECTOR Environment

An intelligent control agent (ICA) based on the H-cell is being implemented to replace the slave agent used in the HECTOR system that helps globally manage the heterogeneous system [10]. The artificial immune based scheduling system is designed around the decentralized decision-making of the ICAs. The ICAs perform all the decision-making functions of the HECTOR system. The ICAs are distributed among the nodes running parallel tasks, monitoring run-time performance and performing task control functions such as starting, migrating, restarting, and notifying tasks. System information such as memory, CPU, network, and disk usage are monitored to determent resource utilization of the node.

The ICAs run on all platforms that are candidates for executing parallel jobs. The primary focus of the ICA is performance optimization of the parallel tasks running on its node. Once every five seconds, the ICA takes performance measurements from the node it is residing on. A five-second interval was selected, because it provides a reasonable compromise between frequent updates and excessive system usage. Maintaining awareness of system performance is important for two reasons. First, it is the means by which the ICAs can detect external load and determine the appropriate action. Second, it permits the agent to identify the machines that are most appropriate to run tasks. Because of the need of constant performance information a new capability has been added to the ICA. This phase, known as the monitoring phase, gathers performance information, and structures it as a string of floating values for input into the intelligent controller portion of the agent.

The system calls used to gather the performance information were used from the original structure that was established for the slave agents used in HECTOR. On an unloaded single-processor Sun 110 MHz Sparc 5 workstation, the ICA uses 2.14 Mbytes of memory. Out of that, 617 Kbytes of it is resident. The measurements increased as the number of tasks increased, and the wall-clock indicated an additional 8.3 ms due to the extra tasks.

6 Conclusions

This paper has described the biological basis for intelligent agents based upon the human immune system. The primary functionalities of the human immune system to be captured in intelligent agents for an artificial immune system are:

- two step process of recognition followed by resolution (similar to content addressable memory)
- distributed responsibility among agents
- communication among agents
- evolutionary adaptation
- immunological memory
- self-organization
- judicious use of resources

Also, it has been proposed that two different types of agents be developed – H-cells for hardware management and S-cells for software. An implementation of this proposed methodology is presently under development.

Reference:

1. Farmer, J. Doyne, Norman H. Packard, and Alan S. Perlson (1986). The Immune System, Adaption, and Machine Learning, *Physica* 22D, pp. 187-204.
2. Hunt, John E. and Denise E. Cooke (1996). Learning Using an Artificial Immune System, *Journal of Network and Computer Applications*, 19, pp 189-212.
3. Roitt, Ivan, Jonathan Brostoff, and David Male (1996). *Immunology* 4th Ed., Mosby.
4. D'haeseleer, P., Stephanie Forrest, and Paul Helman (1996). An Immunological Approach to Change Detection: Algorithms, Analysis, and Implications, *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pp. 110-119.
5. Ishiguro, A., Yuji Watanabe, Toshiyuki Kondo, and Yoshiki Uchikawa (1996). Decentralized Consensus-Making Mechanisms Based on Immune System, *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pp. 82-87.
6. Ishida, Y. and N. Adachi (1996). An Immune Algorithm for Multiagent: Application to Adaptive Noise Neutralization, *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1739-1746.
7. Kephart, Jeffrey O. (1994). A Biologically Inspired Immune System for Computers, *Artificial Life IV: Proceedings of the 4th International Workshop on the Synthesis and Simulation of Living Systems*, pp. 130-139.
8. Kumar, K. K. and James Neidhoffer (1997). Immunized Neurocontrol, *Expert Systems with Applications*, Vol. 13, No. 3, pp. 201-214.
9. Xanthakis, S., S. Karapoulios, R. Pajot, and A. Rozz (1995). Immune System and Fault Tolerant Computing, *Artificial Evolution: European Conference AE 95, Lecture Notes in Computer Science*, Vol.1063, Springer-Verlag, pp. 181-197.
10. Russ, S. R., Jonathan Robinson, Brian K. Flachs, and Bjorn Heckel. The Hector Parallel Run-Time Environment, accepted for publication in *IEEE Transactions on Parallel and Distributed Systems.*

11. Bartfai, Guszti (1994). Hierarchical Clustering with ART Neural Networks, *Proceedings of the IEEE International Conference on Neural Networks*, Vol. 2, IEEE Press, pp. 940-944.

Appendix: Glossary

antibody - a molecule produced by a B cell in response to an antigen that has the particular property of combining specifically with the antigen which induced its formation

antigen - any molecule that can be specifically recognized by the adaptive elements of the immune system (either B or T cells) (originally - antibody generator)

cytokines - the general term for a large group of molecules involved in signaling between cells during immune responses

epitope - molecular shapes on an antigen recognized by antibodies and T cell receptors (i.e., the portion of an antigen that binds with the antibody paratope)

paratope - the specialize portion on the antibody molecule used for identifying other molecules

pathogen - an organism which causes disease