Immunology as Information Processing

Stephanie Forrest
Steven A. Hofmeyr

1 INTRODUCTION

This chapter describes the behavior of the immune system from an information-processing perspective. It reviews a series of projects conducted at the University of New Mexico and the Santa Fe Institute, which have developed and explored the theme "immunology as information processing." The projects cover the spectrum from serious modeling of real immunological phenomena, such as crossreactive responses in animals and the generation of diversity, to computer science applications, especially the attempt to develop an immune system for computers to protect them against viruses, intrusions, and other malicious activities.

In each project, we have used an approach with the following steps: (1) Identify a specific mechanism that appears to be interesting computationally, (2) write a computer program that implements or models the mechanism, (3) study its properties through simulation and mathematical analysis, and (4) demonstrate its capabilities, either by applying the model to a biological question of interest or by showing how it can be used profitably in a computer science setting.

Design Principles for Immune System & Other Distributed Autonomous Systems. L. A. Segel and I. R. Cohen, eds. Oxford Univ. Press, 2000.

2 INFORMATION PROCESSING IN THE IMMUNE SYSTEM

The terms "information processing" and "computation" are not easily defined. For the purposes of this chapter we use the term "information" to refer to a spatio-temporal pattern that can be understood and described independently of its physical realization. We will use the words "computation" and "information processing" interchangeably to describe processes that operate on, or transform, information. In the immune system, we believe that such patterns occur in peptides, proteins, and other molecules and that the recognition, learning, storage, communication, and transformation of these patterns governs the behavior of the immune system. This is a strong claim, one that can be contrasted with the more conventional structural view of immunology, which models cells, molecules, and their interactions as mechanical devices. The following sections illustrate how emphasizing the informational properties of the immune system can provide insights which extend our knowledge beyond that provided by the structural view.

The immune system processes peptide patterns using mechanisms that in some cases correspond closely to existing algorithms for processing information (e.g., the genetic algorithm), and it is capable of exquisitely selective and well-coordinated responses, in some cases responding to fewer than ten molecules. Some of the techniques used by the immune system include learning (affinity maturation of B cells, negative selection of B and T cells, and evolved biases in the germline), memory (crossreactivity and the secondary response), massively parallel and distributed computations with highly dynamic components (on the order of 10^8 different varieties of receptors [52] and 10^7 new lymphocytes produced each day [37]), and the use of combinatorics to address the problem of scarce genetic resources (V-region libraries).

It is generally believed that one major function of the immune system is to help protect multicellular organisms from foreign pathogens, especially replicating pathogens such as viruses, bacteria, and parasites. In order to succeed, the immune system must be capable of distinguishing harmful foreign material (which we will refer to as "nonself") from normally behaving constituents of the organism (which we will label "self").[1] That is, it must be able to recognize foreign material (also called "antigen") as foreign—a problem we cast as one of pattern recognition. Detection of pathogens is accomplished by lymphocytes and antibodies, which function as small independent detectors, circulating throughout the body in the blood and lymph systems. Lymphocytes (B cells and T cells) recognize pathogens by forming molecular bonds

[1] For many years, the problem of immunology was described as that of discriminating self from nonself. However, this view is clearly oversimplified. There are, for example, harmless bacteria (nonself) which are tolerated by the immune system. Likewise, the immune system attacks bona fide self cells, for example, in some cancers. Granting these complexities, and without taking a position in the ongoing debate about the immune system's true function, we will use the self/nonself language to refer to the classes of patterns which the immune system tolerates or tries to eliminate, respectively.

between pathogen fragments and receptors on the surface of the lymphocyte. The more complementary the molecular shape and electrostatic surface charge between pathogen and receptor, the stronger the bond (or the higher the affinity). Thus, we say that, in the immune system, pattern recognition is *implemented* as binding. When an immune system detector (including B cells, T cells, or antibodies) binds to a peptide, we say that the immune system has *recognized* the pattern encoded by the peptide.

We have extracted the following informational design principles from our study of immunology: The immune system is *diverse*, which greatly improves robustness, on both a population and individual level, for example, different people are vulnerable to different pathogens; it is *distributed*, consisting of many components which interact locally to provide global protection, so there is no central control, and hence no single point of failure; it is *error tolerant* in that a few mistakes in classification and response are not catastrophic; it is *dynamic*, i.e., individual components are continually created, destroyed, and are circulated throughout the body, which increases the temporal and spatial diversity of the immune system; it is *self-protecting*, i.e., the same mechanisms that protect the body also protect the immune system itself; and it is *adaptable*, i.e., it can learn to recognize and respond to new pathogens, and it can retain a memory of those pathogens to facilitate future responses.

Here, we focus on three examples of the information-processing perspective: How the immune system distributes its detection through negative selection and other mechanisms, the affinity maturation process and its relation to genetic algorithms, and crossreactive memory and its relation to associative memories.

2.1 HOW THE IMMUNE SYSTEM DISTRIBUTES DETECTION AND RESPONSE

The immune system is distributed throughout the body, and this important architectural feature affects nearly everything else. Certainly, any computational model of an immune system must account for distribution, as it constrains many of the details involving detection, learning, memory, and response. One obvious reason for the importance of distribution in the immune system is that the pathogens to which it must respond are themselves distributed throughout the body. Another reason is that distribution makes the system much more robust to attack—it is difficult to neutralize a system which is not in only one place. Distributed systems have always appealed to computer scientists because of their potential efficiency (by distributing workload over multiple locations) and fault-tolerance (robustness to component failures), but these properties are rarely achieved in artificial systems except in highly specialized settings.

How does the immune system achieve highly distributed memory and control? There are three principle mechanisms: (1) negative selection, (2) costimulation and other signaling events, and (3) cell division and death. Negative

selection allows the immune system to perform self/nonself discrimination with little or no communication between individual detectors (distributed detection). Costimulation and other signaling occurs locally and controls key processes such as cell proliferation, activation, and death. These processes allow the immune system to allocate resources (cells) dynamically in areas where they are most needed and to prevent many autoimmune reactions. Because each cell carries a complete description of itself, these processes also provide vehicles for distributing the information needed for the immune system to function (distributed autonomous control). We believe that each of these mechanisms is crucial to the highly distributed nature of the immune system. In the remainder of this section, we discuss each of these mechanisms in more detail.

One of the principle ways that the immune system achieves self-tolerance (correct discrimination between self and nonself) is by allowing its detector lymphocytes to mature in isolated settings, the thymus in the case of T cells and the bone marrow for B cells. This is necessary because the binding regions for these cells are created through a pseudorandom genetic process, which could easily lead to self-reactive cells. Focusing on T cells, there are several stages of maturation, including genetic rearrangements, positive selection, and negative selection. Of particular interest is negative selection, in which T cells that bind sufficiently strongly with self-proteins expressed in the thymus are destroyed. In addition to negative selection, there is also positive selection of those T cells that can bind weakly to certain molecules. Negative selection prevents T cells from binding to normal self-proteins, while positive selection ensures that T cells will be able to bind to self-cells that express abnormal peptides. T cells that survive both positive and negative selection are allowed to mature, leave the thymus, and become part of the active immune system. Once in circulation, if a T cell binds to antigen in sufficient concentration, a recognition event is said to have occurred, triggering the complex set of events that leads to elimination of the antigen. These negative-selection processes are called *centralized tolerance* because the cells are censored in a single location.

T-cell censoring can be thought of as defining a set (the set of self-peptides) in terms of its complement (all nonself peptides). We can use this principle to design a distributable change-detection algorithm with interesting properties. Suppose we are given a collection of digital data, which we will call "self," and that we wish to monitor self for changes. The data might be an activity pattern, a static program code, or a file of data. The algorithm works as follows: (1) generate a set of detectors which fail to match self (assuming a "closed world," each detector is then guaranteed to match some portion of nonself); (2) use the detectors to monitor the protected data; and (3) record activated detectors. Whenever a detector is activated, a change is known to have occurred, and the location of the change is known (by examining the pattern which activated the detector). There are several details which must be specified before we have an implementable algorithm (1) How are the detectors represented? (2) How is a match defined? (3) How are detectors

generated? (4) How efficient is the algorithm? These topics are explored in detail in Forrest et al. [14] and D'haeseleer [6, 7], but we give highlights here.

There are many possible definitions of self for a computer. Our definitions rely on the idea that any pattern (e.g., a computer program, an execution trace of a program, or the flow of packets through a local area network) can be represented as a finite-length string of symbols. In particular, we represent self as a set of equally sized strings (e.g., by logically segmenting a computer program into equal-length substrings), where the symbol l denotes the length of each string. Similarly, each detector can be defined to be a string of the same length as the substrings.

A perfect *match* between two strings of equal length means that at each location in the string, the symbols are identical. However, perfect matching (perfect binding) is rare in the immune system and improbable between strings of any significant length. Partial matching in symbol strings could be defined in many ways, including Hamming distance or edit distance. However, we typically use a more immunologically plausible rule called r-contiguous bits [38]. This rule looks for r contiguous matches between symbols in corresponding positions. Thus, for any two strings x and y, we say that $match(x, y)$ is true if x and y agree (match) in at least r contiguous locations. The value r is a threshold and determines the specificity of the detector, which is an indication of the number of strings that can be matched (detected) by a single detector. For example, if $r = l$, the matching is completely specific; that is, the detector will detect only a single string (itself). A consequence of a partial matching rule with a threshold, such as r-contiguous bits, is that there is a tradeoff between the number of detectors used, and their specificity. As the specificity of the detectors increases, so the number of detectors required to achieve a certain level of coverage also increases.

Detectors can be generated in several ways. A general method (one that works for any matching rule) is also the one used by the immune system. Simply generate detectors at random, censor them against self, and eliminate those that match self. For the "r-contiguous bits" rule defined above, this generating procedure is inefficient—the number of random strings that must be generated and tested is approximately exponential in the size of self. However, more efficient algorithms based on dynamic programming methods allow us to generate detectors in linear time for the r-contiguous bits rule [6].

The negative-detection algorithm has several interesting properties. First, it can be easily distributed because each detector can function independently of other detectors, that is, without communication between detectors or coordination of multiple detection events. This is because each detector covers part of nonself. A set of detectors can be split up over multiple sites, which will reduce the coverage at any given site but which will provide good syste-mwide coverage. To achieve similar coverage using detectors which match against self would be much more expensive. In this second case (which we call "positive detection"), the system would maintain a description of self and notice when a pattern appeared that failed to match the description. To do this, either a

complete set of the detectors (to specify the complete normal pattern) would be needed at every site, resulting in multiple copies of the detection system, or the sites must be in continual communication to coordinate their results. To see why this is true, consider a pattern that fails to match the positive detectors. The match failure could be for two reasons—either the pattern will be matched by a positive detector located at another location, or it is a true anomaly.

A second point about the negative-selection algorithm is that if we assume a closed world and a complete specification of self, it will never report false positives. Depending on how the detector sets are chosen, however, there is a chance of false negatives. Consequently, the algorithm is likely to be more applicable to dynamic or noisy data where perfect discription is difficult to achieve. This is in contract to cryptographic applications, where the data are static, it is important to detect any one-bit change, and efficient change-detection methods already exist. The number of detectors that is required to detect nonself (using the r-contiguous bits matching rule) depends on how the self set is organized, what false-negative rate we are willing to tolerate, and choice of matching rule [6, 14]. For randomly chosen self sets, the number is roughly the same order of magnitude as the size of self, but for nonrandom data the number is often much lower.

A second mechanism that supports distributed processing in the immune system is the concept of a *second signal*, also known as *costimulation*. From the information-processing perspective, this mechanism helps the immune system avoid autoimmune reactions (or false positives), especially in the presence of distributed learning or adaptation (e.g., somatic hypermutation). It also allows the immune system more flexibility in determining tolerance. Rather than a completely centralized tolerance mechanism (e.g., if all self/nonself determination were performed in the thymus), the second signal is one means by which the immune system can determine tolerance in the periphery, by taking advantage of the fact that self-patterns occur much more frequently than nonself patterns.

As Hofmeyr described [26], one example of how a second signal works is given by T-helper lymphocytes. To review, when a B lymphocyte (that is possibly a mutated descendant of an earlier lymphocyte that survived negative selection) binds a foreign peptide (the first signal), it requires additional stimulation by a signal from a T-helper lymphocyte (that has been censored against self in the thymus) in order to trigger an immune response. The second-signal system thus prevents mutating B-lymphocyte cell lines from incorrectly reacting against self. It also helps prevent autoimmunity in T cells, in the following way. Not all peripheral self-proteins are expressed in the thymus. Consequently, T cells emerging will not necessarily be tolerant of all self-proteins. Self-tolerance in these T cells can also be assured through costimulation. In this case, the first signal occurs when binding exceeds the affinity threshold, but the second signal is provided by the cells of the innate immune system, such as macrophages. We call this *frequency tolerization* [24], because a self-

reactive T cell is likely to encounter self in the absence of tissue damage with much higher frequency than self in the presence of tissue damage; i.e., self is much more frequent than nonself. A similar situation arises in computer security settings where normal behavior is more frequent than malicious intrusions (see section 4). This general signaling strategy is widespread in the immune system, and it can involve more than two signals.

The processes of negative selection and costimulation help the immune system determine which patterns it should tolerate and which patterns it should eliminate. That is, negative selection and costimulation address the problem of representation and learning of representations. The processes of cell replication and death are a third method by which the immune system achieves its distributed organization. However, here the emphasis is on control. Instead of employing a central process to generate and manage all detectors, immune system control is distributed throughout the body. By control, we refer to the immune system processes that allocate resources, determine which type of immune response will be invoked (effector choice), and know how and when to shut down an immune response. Immune cells are self-replicating (which allows the system to be more autonomous and more adaptable), and their control functions result from the processes of programmed cell death, competition for antigen, and so forth.

How might these ideas be useful to computer science? An example is in computer security, where we are concerned about the problem of protecting an artificial immune system from being attacked. If we implemented protective mechanisms as a distributed collection of self-replicating modules, it would be much more difficult to neutralize the entire defense system. There is a strong analogy here to computer viruses, which are a powerful form of distributed computation (unfortunately, to date mostly harmful). A major problem with computer viruses is uncontrolled replication. At most, they check to see if a file has already been infected before reinfecting it. What is needed is a distributed way to regulate the replication and destruction of distributed agents/detectors. Once again, we can take inspiration from the immune system.

We have already described some of the mechanisms through which B cells can be stimulated to clone themselves. For example, when an antigen is recognized in the presence of T-cell help (indicating that the antigen is not part of self), the B cell is stimulated to divide. Unchecked, this would eventually lead to a disproportionate number of B cells of one type. Thus, each increase in cell division must be balanced out at some point by a concomitant reduction in the rate of reproduction and the removal of excess cells through programmed cell death (apoptosis). The dynamic control of relative cell birth and death rates is an important component of the immune system's distributed control system. The immunology behind this control system is complex and only partially understood (for a review, see Boise and Thompson [1]). However, the basic principles have been identified and provide a reasonable starting point. These include costimulation (discussed earlier) and negative feedback

cycles, in which lymphocyte activation promotes immediate survival of cells but also triggers mechanisms that eventually lead to their deletion at the end of the response.

Many different signaling molecules, called cytokines, are believed to participate in the immune response (see Denny [4]), but how they all work together has not been explained systematically [55]. This complex network of signaling molecules and cells apparently has the following properties [41, 42]: (1) every cytokine type affects multiple cells; (2) every function (immune response) is affected by multiple cytokines; (3) immune cells secrete a mixture (vector) of cytokines; (4) signals are molecules, and therefore, distributed (locally) by diffusion; and (5) these signals can be subverted (e.g., viruses can evolve to avoid or interfere with cytokines, perhaps by blocking receptors), so there is an evolutionary pressure toward robust, secure networks. Relevant to computer science, the cytokine signaling networks provide interesting clues about how to design a distributed autonomous control network that is dynamic (both the nodes and connections are changing in time), robust to small perturbations, but responsive to large perturbations.

To summarize, negative selection of detectors provides centralized tolerance and then gives the immune system its ability to distribute detection. Costimulation allows the immune system to distribute its censoring (frequency tolerization). Finally, the processes of cell replication, apoptosis, and the local diffusion of cytokines give the immune system the ability to distribute its resource allocation decisions and control.

2.2 AFFINITY MATURATION AND GENETIC ALGORITHMS

In a *primary response* the immune system uses learning mechanisms similar to biological evolution to design detectors that are specific to a particular antigen. Learning is required if the antigen is "new," that is, if it has not previously been encountered in the lifetime of the organism. Let us consider extracellular pathogens, against which B cells are a primary defense. When a B cell is activated by binding pathogen, it produces many copies of itself (clones that are produced through cell division), in a process called *clonal expansion*. The resulting cells can undergo *somatic hypermutation*, creating daughter B cells with mutated receptors. These new B cells compete for pathogens with their parents and with other clones. The higher the affinity of a B cell for available pathogens, the more likely it is to clone. This results in a Darwinian process of variation and selection, called *affinity maturation*. Affinity maturation enables B cells to adapt rapidly to the specific pathogens present in the body. High-affinity B cells deal with pathogens efficiently by secreting antibody, which can promote pathogen destruction. This is especially important when the immune system is fighting off a replicating pathogen, a situation which is essentially a race between pathogen reproduction and B-cell reproduction. Efficient binding of pathogens is also required to clear infections completely.

The affinity maturation process is reminiscent of a genetic algorithm [10, 16, 27], but there are some important differences. In their traditional form, genetic algorithms process a population of individuals. The population is created randomly in the initial generation. In the simplest case, each individual is a bit string, typically representing a candidate solution to a problem. Variations among individuals in the population result in some individuals being more fit than others (e.g., better problem solutions). These differences are used to bias the selection of a new set of individuals at the next time step, referred to as selection. During selection, a new population is created by making copies of more successful (more fit) individuals and deleting less successful ones. However, the copies are not exact. There is a probability of mutation (random bit flips), crossover (exchange of corresponding substrings between two individuals), or other changes to the bit string during the copy operation. By transforming the previous set of good individuals to a new one, the genetic operations generate a new set of individuals, which have a better than average chance of having high fitness. When this cycle of evaluation, selection, and genetic operations is iterated for many generations, a population of individuals arises, that is biased towards highly fit individuals.

Genetic algorithms can be viewed as a first-order model of the affinity maturation process, as well as a model of change over evolutionary time scales. In the affinity maturation case, each individual in the population can be thought of as a single B cell. High-fitness B cells are those that bind with high affinity to frequently occurring antigen, and they are activated to produce B-cell clones (selection). Somatic hypermutation is the genetic operator, that allows the population of B-cell clones to evolve to be highly specific to the frequently occurring antigen. Over time, a population of B cells is produced that binds much more tightly to the prevalent antigen than before the process started. There are two important differences between affinity maturation in the immune system and conventional genetic algorithms. First, in an immune system undergoing hypermutation, there is no obvious equivalent to crossover. In this case, we can think of the crossover probability being set to zero. A second difference is that in the immune system the amplification (selection) and mutation phases are apparently distinct [31], whereas in the genetic algorithm they are interleaved.

As we will see in section 3.1, the genetic algorithm can be used to study different hypotheses about how the immune system is likely to have evolved. In this case, each individual in the genetic algorithm corresponds to a biological individual (more properly, the immune system genes of a biological individual), fitness corresponds to the survivability of that individual against pathogens, and mutation and crossover correspond to the genetic changes that an individual passes on to his or her offspring.

2.3 CROSSREACTIVITY AND ASSOCIATIVE MEMORIES

A successful immune response results in the proliferation of B cells that have high affinities for the foreign pathogens that caused the response. The information encoded in these B cells constitutes the "memory" of the immune system. Understanding immune memory is problematic because B cells typically live for just a few days, and once an infection is eliminated, it is not well understood what prevents the adapted subpopulation of B cells from dying out. On subsequent encounters with the antigen, however, the immune system responds with a *secondary response* in which the memory cells for the earlier antigen quickly produce large quantities of specific antibodies. These secondary responses are much faster and stronger than primary responses. The distinction between primary and secondary responses is the basis for vaccination, and it is the reason why we get diseases such as measles only once. In vaccination, an attenuated version of the antigen is injected to prime the immune system, so that when the real antigen is encountered, it can produce a response quickly and in large volume.

B-cell receptors do not require an exact match to an antigen in order to be activated. If the immune system is primed with a particular antigen, and then presented with a related antigen (one that is structurally similar but not exactly the same) some of the memory components for the first antigen can be stimulated by the second antigen, producing a secondary immune response. Thus, it is sometimes possible to produce a secondary response to an antigen that the immune system has never seen before. In the field of associative memories, this is called association or generalization [49]. In immunology, such a secondary response is called "crossreactive memory," or "original antigenic sin." Crossreactivity can be beneficial (as in the case of vaccinating with cowpox to protect against the related disease smallpox), and it can be harmful (as in the case where a secondary response is ineffective at eliminating the new antigen but blocks an effective primary response).

3 IMMUNE SYSTEM MODELING

In section 2 we discussed immunology from an informational perspective. Here, we describe how such a perspective can contribute to immune system modeling. The models we describe are all based on a universe in which antigens and detectors are represented by strings over a small alphabet of symbols, and interactions among strings represent molecular binding. In effect, we represent only the receptor region on the surface of a lymphocyte. This approach, first introduced by Farmer et al. [8], is now now widely adopted in the theoretical immunological community. It has been used to study a wide variety of immune system mechanisms. For a recent example see Detours and Perelson [5]. In these "artificial immune systems," binding takes place when an antibody

string and an antigen string have similar binary patterns.[2] Binding between idealized antibodies and antigens is defined by a matching function that rewards more specific matches over less specific ones, as we saw earlier with the r-contiguous bits rule. This constraint is related to the immune system's ability to distinguish self from nonself, because recognition of nonself must be fairly specific in order to avoid recognizing self.

Representing binding between antigens and antibodies as simple string matching has advantages and disadvantages. On the one hand, it fails to capture much of what is known and of interest to immunologists—the details of specific molecular and cellular interactions. On the other hand, it provides the ability to model many cells and their interactions, something that we think will be increasingly important over the next several years, both for understanding immune system function and for exploiting immune system principles in computing. Also, it allows us to isolate the informational aspects of immune system processing from other potential factors such as dose, antigenicity, immunocompetence, and virulence and transmissibility, as we will see in section 3.2.

With this basic modeling abstraction in place we can construct a wide variety of model immune systems. For example, affinity maturation can be modeled in such a system by constructing one population of antibodies and one of antigens, each from bit strings. Antigens are "presented" to the antibody population one at a time, and high-affinity antibodies have their fitness increased. The antibody population is then evolved by a genetic algorithm based on its success at matching antigens [13, 50]. Similarly, crossreactive memory can be modeled by constructing populations of B cells, presenting the system with a single antigen type, allowing affinity maturation, then presenting the system with one or more related antigens, and studying the strength of the response [47]. The following subsections describe two such models, one that is concerned with the evolutionary pressures that shaped the immune system and one that is concerned with the question of repeated vaccination against a mutating virus.

3.1 DIVERSITY GENERATING MECHANISMS

One question that can be addressed with models such as these involves the generation of diversity. Several mechanisms have been discovered by which the immune system is able to generate its enormously diverse set of receptors. These include: immune receptor libraries [43], combinatorial rearrangement of entries from multiple libraries [52], junctional diversity [15], and somatic hypermutation and/or gene conversion [54]. We would like to better understand the relative contributions of these different mechanisms, and if possible, what role each mechanism plays in generating the immune repertoire. A re-

[2] A more direct analogy with binding in immunology would be based on complementary binary patterns. In binary alphabets, however, matching rules based on complementary bit patterns are logically equivalent to those based on similarity.

lated question is whether the immune system has evolved its genetics to cover pathogen space randomly or if it has evolved biases that incorporate learning about the pathogenic environment. These questions are difficult to address experimentally in animals.

Ron Hightower's 1996 dissertation [18] studied the mechanisms through which the immune system creates its large number of unique receptors—the multigene families. His work provided insight about how and why the natural immune system evolved as it did [40], showed that robust pattern recognizers can be learned with a surprisingly small amount of information [19], and gave an interesting example of a genotype-to-phenotype map that is nontrivial but still quantifiable. His model inserted an interpretation step between the representation manipulated by evolution (the genetic libraries) and the representation that is operative during an individual's life (the expressed antibodies) [20].

Mihaela Oprea then used Hightower's multigene family model as part of a detailed study of the sources and evolutionary significance of diversity in the immune system [34, 35, 36]. Part of this work was a study of how germline diversity (that is, the antibody gene libraries) contribute to the structure of the antibody repertoire. To study the effect of germline diversity, Oprea used a genetic algorithm to model how the immune system might have evolved. (Note, this is a different use of the genetic algorithm from that described earlier for affinity maturation.) The genetic algorithm employed a population of M individuals, called hosts, which evolved in an environment of hostile pathogens. Each pathogen was represented as a bit string. Each individual in the population consisted of an antibody library, containing A antibodies, where each antibody was represented as a bit string of length L (typically, $L = 16$). Pathogens were also represented as bit strings of length L, and the antibody set was evolved on a fixed, but large (2^9), pathogen set. The A antibodies were concatenated to form a single chromosome. This representation of antibody libraries is reminiscent of the so-called "Pitt" approach to classifier systems [2, 28], in which candidate sets of production rules are represented by concatenating the individual rules to form a single chromosome. Good rule sets (those that solve a given problem well) are then evolved using the genetic algorithm. In the case of Oprea's model, each library (one individual's genome) corresponds to a classifier system if we consider each encoded antibody to take the role of a single classifier rule. This aspect of her representation seems to correspond quite directly to V-region genes in humans.

Oprea then used the model to determine how the fitness of a single individual scales with the amount of diversity (number of entries) and with the matching rule used to determine bond strength. Her results suggest that adding more and more antibodies to the genome-encoded repertoire improves the survival probability of the individual by smaller and smaller amounts, the exact relation being determined by the binding rule. She experimented both with binding rules based on Hamming distance [34] and on free-energy calculations [35]. Theses results suggest an explanation for why the V-region

libraries in various species do not seem to number more than approximately one hundred genes. However, if the selection pressure for increasing library size is small, what would keep evolution from producing even smaller libraries than the ones that we observe? One possible explanation is that there is a hard threshold in antibody/pathogen binding, below which recognition will not occur at all. In this case, some minimal number of antibodies would be required to ensure that at least one has minimal affinity for any given pathogen. Alternatively, one can imagine that the pathogen set is structured as a distribution of clusters, such that different antibodies in the library would reflect different clusters of pathogens. Oprea subscribes to the second alternative, conjecturing that the antibody genes encode antibodies which are "strategically" placed in the space of possible receptors, thus providing a form of "coarse graining" of the pathogen space.

3.2 VACCINE DESIGN FOR MUTATING VIRUSES

A second example of how ideas about information processing can be used to better understand the immune system is Derek Smith's 1997 dissertation on the crossreactive immune response and its application to the problem of vaccine design for mutating viruses [48]. Smith developed a model of crossreactive memory in immunology (closely related to Kanerva's Sparse Distributed Memory [30]) and applied the model to the problem of vaccine design for mutating viruses, focusing on influenza [44, 45, 46, 49].

Smith's computational model is concerned with B cells, plasma cells, antibodies, memory B cells, and antigens. The model is *individual based* in the sense that each individual immune cell and antibody is represented in the computer explicitly. (By contrast, many models represent each type of constituent explicitly, together with the concentration in which they exist.) Smith's model is a particularly simple and elegant example of the individual-based style of modeling. Each individual's receptor region is represented as a simple string of twenty symbols, each symbol chosen from a four-letter alphabet (e.g., a, b, c, d). The four symbols are intended to represent generic properties of the binding region; e.g., it might be reasonable to interpret them as polar, nonpolar, large, small. In an earlier study, Smith found that with a binding rule based on the idea of Hamming distance (at how many positions in the string do the symbol strings differ in value?) on strings of length 20 defined over an alphabet of size 4, he obtained crossreactivity patterns that agree well with known biological data [46].

In the model, large populations (10^7) of B cells, each with a randomly generated receptor, are created. The population is then presented with antigen, and the B cells that bind to the antigen with sufficient affinity, under the Hamming rule described earlier, have a chance to be stimulated to divide, undergo somatic hypermutation, and differentiate into a plasma or memory cell. Secreted antibody has a chance to bind antigen, and antigen-antibody complexes are removed from the simulation. Details about how the model is

implemented, using a technique called lazy evaluation, are given in Smith et al. [44].

Smith used his model to study the effect of repeated vaccination against a mutating virus [45]. Influenza is an example of such a virus. Antigenic drift causes new but related strains of the virus to circulate through human populations on an annual basis. Influenza vaccines are, therefore, updated regularly to track the antigenic drift. Smith was interested in the case of human populations who are vaccinated annually with the updated strains. Through a series of simulations, based on historical records of actual observed antigenic distances, he observed cases of positive interference between vaccines (where the first year's vaccine reinforced the effect of the second year's vaccine) and negative interference (where the first year's vaccine prevented the second year's vaccine from being effective). He analyzed these cases in detail and concluded that the different outcomes resulted from specific combinations of antigenic distances between the first-year vaccine, the second-year vaccine, and the epidemic strain that actually appears after the second-year vaccine is administered. These different cases are illustrated in figure 1.

Smith then compared his simulation results with the published epidemiological studies on vaccine efficacy (see fig. 2). As the figure shows, Smith's simulation results agree closely with the epidemiological studies, even though his model is exquisitely simple. His results are significant because they provide a parsimonious explanation, based on antigenic distances, for both the cases in which multiyear vaccines are successful and the cases in which they are not. Further, his explanation relies only on the informational properties of the receptors (how closely they are related to each other in sequence space) and not on biological details of how the receptors are implemented, deployed, or on biological properties of the virus. Finally, his dissertation makes testable predictions about how different antigenic strains might interact with one another and with the wild-type strain encountered during an epidemic.

4 IDEAS FROM IMMUNOLOGY APPLIED TO COMPUTER SECURITY

The immune system can also be studied for the purpose of designing better artificial adaptive systems. A natural domain in which to apply immune system mechanisms is computer security, where the analogy between protecting the body and protecting a normally operating computer is evident. A computer security system should ensure the *integrity* of a machine or set of machines, protecting them from unauthorized intruders and foreign code. This is similar in functionality to the immune system protecting the body (*self*) from invasion by harmful microbes (*nonself*). Within this domain, we have studied several problems, including computer virus detection [6, 14], host-based intrusion detection [11, 12, 22, 53], automated response [51], and network intrusion detection [23, 25]. This last project incorporates several different immune sys-

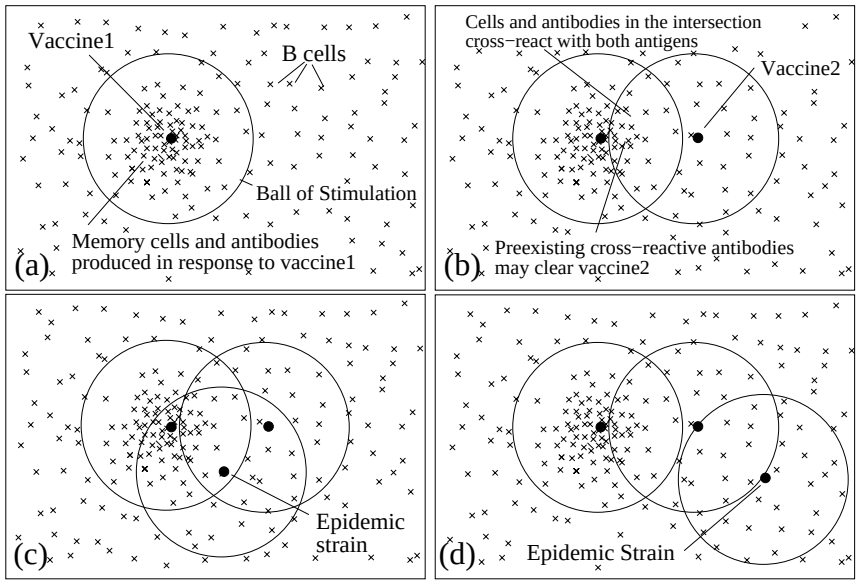

FIGURE 1 An illustration of the antigenic distance hypothesis. *Shape space* diagrams are a way to illustrate the affinities between multiple B cells/antibodies and antigens, and also the antigenic distances between antigens [39]. In these shape space diagrams, the affinity between a B cell or antibody (X) and an antigen (•) is represented by the distance between them. Similarly, the distance between antigens is a measure of how similar they are antigenically. (a) B cells with sufficient affinity to be stimulated by an antigen lie within a *ball of stimulation* centered on the antigen. Thus, a first vaccine (vaccine 1) creates a population of memory B cells and antibodies within its ball of stimulation. (b) Crossreactive antigens have intersecting balls of stimulation, and antibodies and B cells in the intersection of their balls—those with affinity for both antigens—are the crossreactive antibodies and B cells. The antigen in a second vaccine (vaccine 2) will be partially eliminated by preexisting crossreactive antibodies (depending on the amount of antibody in the intersection), and thus the immune response to vaccine 2 will be reduced [3, 9]. (c) If a subsequent epidemic strain is close to vaccine 1, it will be cleared by preexisting antibodies. (d) However, if there is no intersection between vaccine 1 and the epidemic strain, there will be few preexisting crossreactive antibodies to clear the epidemic strain quickly, despite two vaccinations. Note, in the absence of vaccine 1, vaccine 2 would have produced a memory population and antibodies that would hve been protective against both the epidemic strains in (c) and (d). For an antigen with multiple epitopes (such as influenza), there would be a ball of stimulation for each epitope. Printed with permission by *PNAS* **96** (1999). National Academy of Sciences, USA.

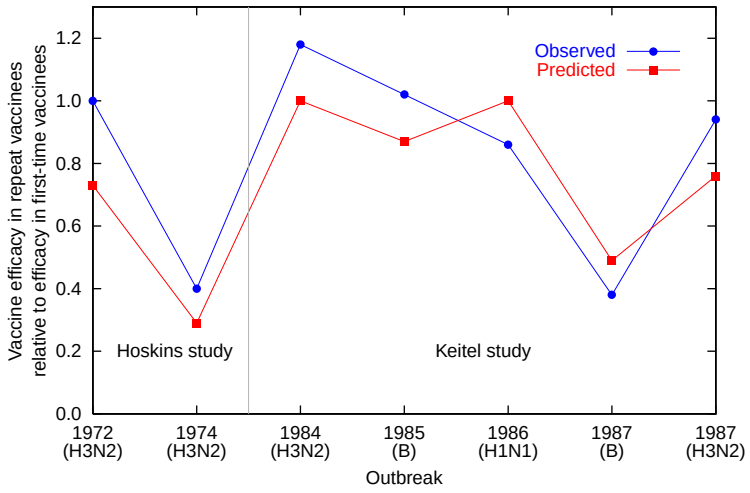

FIGURE 2 Observed vaccine efficacy in repeat vaccinees relative to the efficacy in first-time vaccines, and predicted vaccinees efficacy based on the antigenic distance hypothesis. Printed with permission by *PNAS* **96** (1999). National Academy of Sciences, USA.

tem mechanisms into an integrated system and is the focus of this section, which is largely excerpted from Hofmeyr and Forrest [21].

Hofmeyr's network immune system is situated in a local-area broadcast network (LAN) and is used to protect the LAN from network-based attacks. In contrast with switched networks, broadcast LANs have the convenient property that every location (computer) sees every packet passing through the LAN. In this domain, self is defined to be the set of normal pairwise connections (at the TCP/IP level) between computers, including connections between two computers in the LAN as well as connections between one computer in the LAN and one external computer (fig. 3). A connection is defined in terms of its "data-path triple"—the source IP address, the destination IP address, and the service (or port) by which the computers communicate [17, 33]. This information is compressed to a single 49-bit string that unambiguously defines the connection. Self is then the set of normally occurring connections observed over time on the LAN, each connection being represented by a 49-bit string. Similarly, nonself is also a set of connections (using the same 49-bit representation), the difference being that nonself consists of those connections, potentially an enormous number, that are not normally observed on the LAN.

Natural immune systems consist of many different kinds of cells and molecules. Here, we simplify by introducing one basic type of detector cell which combines useful properties from several different immune cells. This

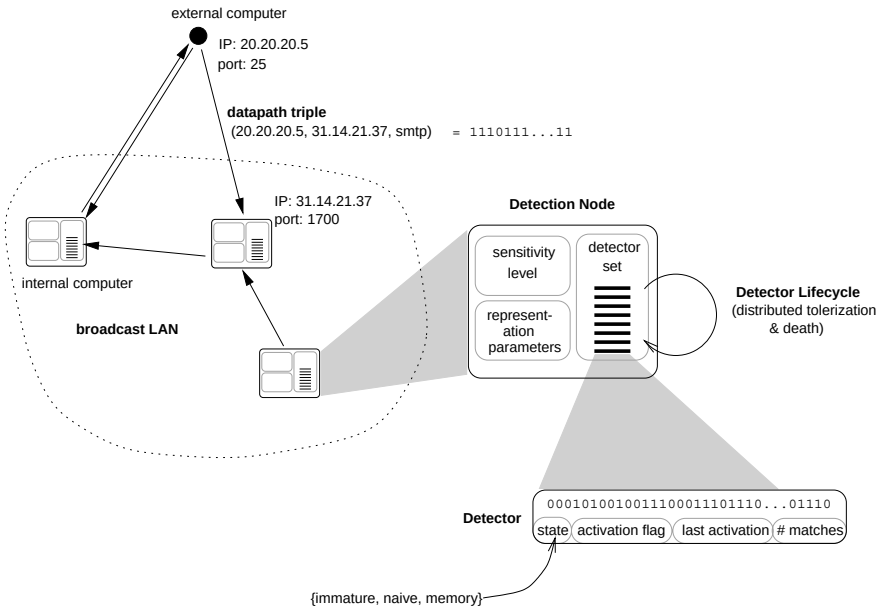

FIGURE 3 Architecture of the artificial immune system.

detector cell has several different possible states, which correspond roughly to thymocytes (immature T lymphocytes undergoing negative selection in the thymus), naïve B lymphocytes (which have never matched foreign material), and memory B lymphocytes (which are long-lived and easily stimulated). The natural immune system also has many different types of effector cells, which implement different immune responses (e.g., macrophages, mast-cells, etc.). This set of features was not included in Hofmeyr's model.

Each detector cell is represented by a single bit string of length $l = 49$ bits, and a small amount of local state (see fig. 3). There are many ways of implementing the detectors; for example, a detector could be a production rule, or a neural network, or an agent. We chose to implement detection (binding) as *string matching*, where each detector is a string d, and detection of a string s occurs when there is a match between s and d, according to a *matching rule*. We use string matching because it is simple and efficient to implement, and easy to analyze and understand. Recall that two strings d and s match under the r-contiguous bits rule if d and s have the same symbols in at least r contiguous bit positions.

The detectors are grouped into sets, one set per machine, or computer, on the LAN; each computer loosely corresponds to a different location in the

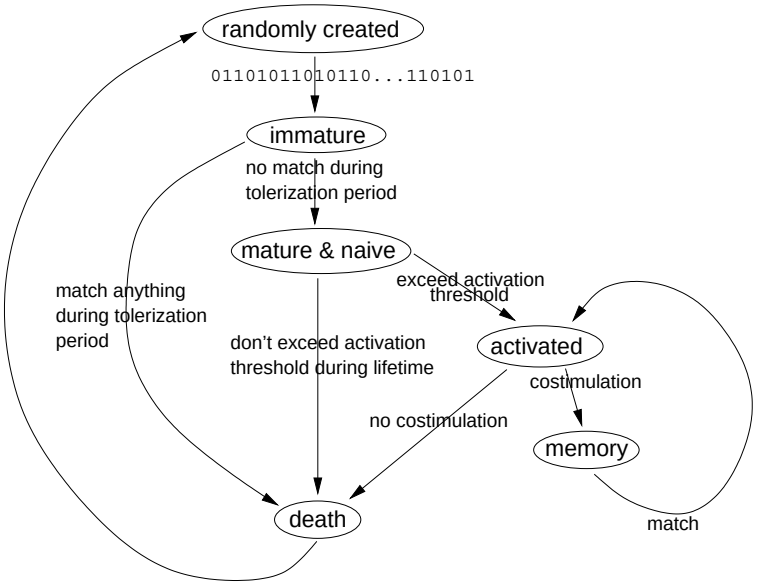

FIGURE 4 Life cycle of a detector. A detector is initially randomly created, and then remains immature for a certain period of time, which is the tolerization period. If the detector matches any string a single time during tolerization, it is replaced by a new randomly generated detector string. If a detector survives immaturity, it will exist for a finite lifetime. At the end of that lifetime it is replaced by a new random detector string, unless it has exceeded its match threshold and becomes a memory detector. If the activation threshold is exceeded for a mature detector, it is activated. If an activated detector does not receive costimulation, it dies (the implicit assumption is that its activation was a false positive). However, if the activated detector receives costimulation, it enters the competition (see above) to become a memory detector with an indefinite lifespan. Memory detectors need only match once to become activated.

body.[3] Because of the broadcast assumption, each detector set is constantly exposed to the current set of connections in the LAN, which it uses as a dynamic definition of self (i.e., the observed connections in a fixed time period are analogous to the set of proteins expressed in the thymus during some period of time). Within each detector set, new detectors, or thymocytes, are created randomly and asynchronously on a continual schedule, similarly to the natural immune system. These new detectors remain *immature* for some period of time, during which they have the opportunity to match any current

[3]The ability of immune system cells to circulate throughout the body is an important part of the immune system that we are currently ignoring. In our system, detectors remain in one location for their lifetime.

network connections. If a detector matches when it is immature, it is killed (deleted). This process, described earlier, is called *negative selection*.

Detectors that survive the initial censoring are promoted to *mature* detectors (analogous to T lymphocytes leaving the thymus and B lymphocytes leaving the bone marrow). Each mature detector can now act independently. If a mature detector d matches a sufficient number of packets (see the discussion of activation thresholds below), an alarm is raised. The time for which d is in the naïve phase can be thought of as a learning phase. At the end of this learning phase, if d has failed to match a packet, it is deleted, but if it has matched a sufficient number of nonself packets then it enters a competition with other activated detectors to become a memory detector. Memory detectors have a greatly extended, potentially infinite, lifetime. Memory detectors have a lower threshold of activation (see below), thus implementing a "secondary response" that is more sensitive and responds more aggressively than naïve detectors to previously seen strings. Although these memory detectors are desirable, a large fraction of naïve detectors must always be present, because the naïve detectors are necessary for the detection of novel foreign packets.

Both the natural immune system and our artificial immune system face the problem of "incomplete self sets." When T lymphocytes undergo negative selection in the thymus, they are exposed to most, but not all, of the proteins in the body. Consequently, the negative selection process is incomplete in the sense that a lymphocyte could survive negative selection but still be reactive against a legitimate self-protein (one that was not presented in the thymus), potentially leading to an autoimmune reaction. In our artificial immune system, such an auto-immune reaction is called a *false positive*. False positives arise if we train the system on an incomplete description of self, and then encounter new but legitimate patterns. We would like the system to be tolerant of such minor, legitimate new patterns, but still detect abnormal activity. We have implemented two methods designed to overcome this problem: activation thresholds and adaptive thresholds. *Activation thresholds* are similar in function to avidity thresholds in lymphocytes. A lymphocyte is covered with many identical receptors, and it is only activated when sufficiently many receptors are bound to pathogens, i.e., when the avidity threshold for binding is exceeded. Analogously, each detector in the artificial immune system must match multiple times before it is activated. Each detector records the number of times it matches, and it raises an alarm only when the number of matches exceeds the activation threshold, which is stored locally for each detector set. Once a detector has raised an alarm, it returns its match count to zero. This mechanism has a time horizon: over time the count of matches slowly returns to zero. Thus, only repeated occurrences of structurally similar and temporally clumped strings will trigger the detection system.

However, some attacks may be launched from many different machines, in which case the first method is unlikely to be successful. To detect such distributed coordinated attacks, we introduce a second method, called *adap-*

tive activation (see fig. 3). Whenever the match count of a detector goes from 0 to 1, the local activation threshold for the set of detectors on a computer is reduced by one. Hence, each different detector that matches for the first time "sensitizes" the detection system, so that all detectors on that machine are more easily activated in future. This mechanism also has a time horizon; over time, the activation threshold gradually returns to its default value. This method will detect diverse activity from many different sources, provided the activity happens within a certain period of time. This mechanism roughly captures the role that inflammation, cytokines, and other molecules play in increasing or decreasing the sensitivity of individual immune system lymphocytes within a physically local region.

Two simple adaptive mechanisms used in our artificial immune system are negative selection and the maturation of naïve cells into memory cells. A third adaptive mechanism has also been incorporated—affinity maturation. In its simple form, detectors compete against one another for foreign packets, just as lymphocytes compete to bind foreign antigen, as described in section 2.2. In the case where two detectors simultaneously match the same packet, the one with the closest match (greatest fitness) wins. This introduces pressure for more specific matching into the system, causing the system to discriminate more precisely between self and nonself. We propose, although we have not yet implemented this, that successful detectors (those that bind many foreign packets) will undergo proliferation (making copies and migrating to other computers) and somatic hypermutation (copying with a high mutation rate).

In our system, we use a human as the second signal. When a detector raises an alarm, there is some chance that it is a false alarm (autoimmune reaction). Before taking action, the artificial immune system waits a fixed amount of time (say, 24 hours) for a costimulatory signal, which in the current implementation is an e-mail message from a human. If the signal is received (confirming the anomaly), the detector enters the competition to become a memory detector with an indefinite lifespan (see above), but, if it loses the competition, it remains naïve and has its match count reset to 0. If the second signal is not received, the artificial immune system assumes that it was a false alarm and destroys the detector, as in the natural immune system. The complete lifecycle of a detector is shown in figure 3.

It might seem more natural to send messages to the artificial immune system in the case of false alarms instead of true anomalies, so that the artificial immune system can adjust itself appropriately by immediately deleting the autoreactive detectors. Unfortunately, this would create a vulnerability, because a malicious adversary could send signals to the artificial immune system, labeling true foreign packets as false alarms, thus tolerizing the artificial immune system against certain forms of attack. The form of costimulation that we have used is much more difficult to subvert. Because false alarms are generally much more frequent than true anomalies, our costimulation method has the additional advantage that action by the human operator is required in the less frequent case.

Each of the mechanisms described above can be implemented with a single detector set running on a single location. We could trivially gain efficiency advantages by distributing the single detector set across all locations on the LAN, thus distributing the computational cost of intrusion detection. Such distribution will give linear speedup, because there are no communication costs (apart from the signaling of alarms and costimulation). However, we take advantage of another immune system feature to implement a more powerful form of distribution.

Molecules of the *major histocompatibility complex* (MHC) play an important role in immune systems, because they transport protein fragments (called peptides) from the interior regions of a cell to its surface, *presenting* these peptides on the cell's surface. This mechanism enables roving immune system cells to detect infections in cells without penetrating the cell membrane. There are many variations of MHC, each of which binds a different class of peptides. Each individual in a population is genetically capable of making a small set of these MHC types (about ten), but the set of MHC types varies in different individuals. Consequently, individuals in a population are capable of recognizing different profiles of peptides, providing an important form of population-level *diversity.*[4] Our artificial immune system uses *permutation masks* to achieve a similar kind of diversity. A permutation mask defines a permutation of the bits in the string representation of the network packets. Each detector set has a different, randomly generated, permutation mask. One limitation of the negative-selection algorithm as originally implemented is that it can result in undetectable abnormal patterns called *holes*, which limit detection rates [6, 7]. Holes can exist for any symmetric, fixed-probability matching rule, but by using permutation masks we effectively change the match rule on each host, and so overcome the hole limitation. Thus, the permutation mask controls how the network packet is presented to the detection system, which is analogous to the way different MHC types present different sets of peptides on the cell surface.

Our network intrusion detection system was empirically tested on actual data collected on a subnet of 50 computers at the Computer Science department at the University of New Mexico. The data consisted of two months of network traffic. This was used as the basis for a simulation of a network of 50 computers. We chose to simulate the environment because we needed to repeat many different runs of the simulation to test out the effects of the various mechanisms. We also collected seven traces of network traffic during real incidents of attempted and successful intrusions (for a description of these intrusions, see Hofmeyr [25]). In the simulation, with each of the 50 computers running with 100 detectors, the false positive rates were on the order of two per day. This is regarded as very low in the intrusion detection community [32]. In addition, the system successfully detected all seven intrusive incidents, in all cases detecting at least 44% of the nonself strings present

[4] For example, there are some viruses, such as the Epstein-Barr virus, that have evolved dominant peptides that cannot be bound by particular MHC types, leaving individuals who have those MHC types vulnerable to the disease [29].

in each trace. The various mechanisms were found to be useful: Activation thresholds reduce false positives by up to a factor of 10; the sensitivity mechanism is useful for detecting distributed coordinated attacks; costimulation reduces false positives by up to a factor of three; memory detectors greatly improve the secondary response; and, diverse permutation masks are useful for detecting anomalies that are similar to the normal traffic.

5 CONCLUSIONS

In this chapter, we emphasized three themes: (1) understanding the distributed memory and control systems of the immune system from an informational perspective, (2) creating models that emphasize the informational properties of the immune system, and (3) building an integrated adaptive immune system that can address an important unsolved problem in computer science.

What is the value of such analogies? In the case of modeling the real immune system, there are several benefits to the information-based approach. As we saw in the influenza example, viewing biological processes as computations can lead to nonobvious predictions that can be tested. It can also allow us to infer global properties of the immune system that are impossible to test experimentally using today's technology. An example of this is Oprea's conclusions about pathogen space coverage, based on her genetic algorithm simulation. Further, simulations allow us the opportunity to perform perturbation experiments and to run controls that may be difficult to do experimentally. Oprea's scaling relations are the result of such an exercise.

In the case of computer science, our study of the immune system has revealed an important set of design principles, many of which are widely appreciated in computer science, but some of which are not. More importantly, we have few examples of working computer systems that illustrate all of these properties in an integrated whole. Hofmeyr's artificial immune system is, however, an important step in this direction. As our computers and the software they run become more complex and interconnected, properties such as robustness, flexibility and adaptability, diversity, self-reliance and autonomy, and scalability can only become more important to computer design.

The research described here stresses the similarities between computers and immunology. Yet, there are also major differences, which it is necessary to respect. The success of all analogies between computing and living systems will ultimately rest on our ability to identify the correct level of abstraction—preserving what is essential from an information-processing perspective and discarding what is not. In the case of immunology, this task is complicated by the fact that real immune systems handle data that are very different from that handled by computers. In principle, a computer vision system or a speech-recognition system would take as input the same data as a human does (e.g., photons or sound waves). In contrast, regardless of how successful we are

at constructing a computer immune system, we would never expect or want it to handle pathogens in the form of actual living cells, viruses, or parasites. Thus, the level of abstraction for computational immunology is necessarily higher than that for computer vision or speech, and there are more degrees of freedom in selecting a modeling strategy.

ACKNOWLEDGMENTS

Alan Perelson introduced the authors to immunology and made significant contributions to most of the projects described in this chapter. Lee Segel is an active collaborator in our ongoing efforts to understand the response side of the immune system. Over the past ten years, an unusually talented and dedicated group of students and postdoctoral fellows have comprised the "adaptive" group at UNM. This chapter reflects their achievements and articulates some of the insights that were hammered out in our weekly seminar.

Forrest is affiliated with the Santa Fe Insitute and the University of New Mexico, Department of Computer Science. Hofmeyr conducted the work described in this chapter while he was a graduate student and post-doctoral fellow at the University of New Mexico. The authors gratefully acknowledge the support of the National Science Foundation (grants IRI-9711199 and CDA-9503064), the Office of Naval Research (grant N00014-99-1-0417), and the Intel Corporation.

REFERENCES

[1] Boise, Lawrence H., and Craig B. Thompson. "Hierarchical Control of Lymphocyte Survival." *Science* **274** (1996): 67–68.

[2] Booker, L. B., R. L. Riolo, and J. H. Holland. "Learning and Representation in Classifier Systems." *Art. Intel.* **40** (1989): 235–282.

[3] Davenport, F. M., A. V. Hennessy, and T. Francis. *J. Exp. Med.* **98** (1953): 641–656.

[4] Denny, T. N. "Cytokines: A Common Signaling System for Cell Growth, Inflammation, Immunity, and Differentiation." This volume.

[5] Detours, V., and A. S. Perelson. "Explaining High Alloreactivity as a Quantitative Consequence of Affinity-Driven Thymocyte Selection." *Proc. Natl. Acad. Sci.* **96** (1999): 5153–5158.

[6] D'haeseleer, P., S. Forrest, and P. Helman. "An Immunological Approach to Change Detection: Algorithms, Analysis, and Implications." In *Proceedings of the 1996 IEEE Symposium on Computer Security and Privacy.* Los Alamitos, CA: IEEE Press, 1996.

[7] D'haeseleer, Patrik. "An Immunological Approach to Change Detection: Theoretical Results." In *Proceedings of the 9th IEEE Computer Secu-*

rity Foundations Workshop. Los Alamitos, CA: IEEE Computer Society Press, 1996.

[8] Farmer, J. D., N. H. Packard, and A. S. Perelson. "The Immune System, Adaptation, and Machine Learning." *Physica D* **22** (1986): 187–204.

[9] Fazekas de St. Groth, S., and R. G. Webster. "Disquisitions on Original Antigenic Sin. II. Proof in Lower Creatures." *J. Exp. Med.* **124** (1966): 347–361.

[10] Forrest, S. "Genetic Algorithms: Principles of Adaptation Applied to Computation." *Science* **261** (1993): 872–878.

[11] Forrest, S., S. Hofmeyr, and A. Somayaji. "Computer Immunology." *Comm. ACM* **40(10)** (1997): 88–96.

[12] Forrest, S., S. Hofmeyr, A. Somayaji, and T. Longstaff. "A Sense of Self for Unix Processes." In *Proceedings of the 1996 IEEE Symposium on Computer Security and Privacy*. Los Alamitos, CA: IEEE Press, 1996.

[13] Forrest, S., B. Javornik, R. Smith, and A. Perelson. "Using Genetic Algorithms to Explore Pattern Recognition in the Immune System." *Evol. Comp.* **1(3)** (1993): 191–211.

[14] Forrest, S., A. S. Perelson, L. Allen, and R. Cherukuri. "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.

[15] Gilfillan, S., A. Dierich, M. Lemeur, C. Benoist, and D. Mathis. "Mice Lacking TdT: Mature Animals with an Immature Lymphocyte Repertoire." *Science* **261** (1993): 1175–1178.

[16] Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison Wesley, 1989.

[17] Heberlein, L. T., G. V. Dias, K. N. Levitte, B. Mukherjee, J. Wood, and D. Wolber. "A Network Security Monitor." In *Proceedings of the IEEE Symposium on Security and Privacy*. Los Alamitos, CA: IEEE Press, 1990.

[18] Hightower, R. "Computational Aspects of Antibody Gene Families." Ph.D. diss., University of New Mexico, Albuquerque, NM, 1996.

[19] Hightower, R., S. Forrest, and A. S. Perelson. "The Evolution of Emergent Organization in Immune System Gene Libraries." In *GACONF6*, edited by L. J. Eshelman. Los Altos, CA: Morgan-Kaufmann, 1995.

[20] Hightower, R. H., S. Forrest, and A. S. Perelson. "The Baldwin Effect in the Immune System: Learning by Somatic Hypermutation." In *Adaptive Individuals in Evolving Populations*, edited by R. K. Belew and M. Mitchell, 159–167. Santa Fe Institute Studies in the Sciences of Complexity. Reading, MA, Addison-Wesley, 1996.

[21] Hofmeyr, S., and S. Forrest. "Immunity by Design: An Artificial Immune System." In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 1289–1296. San Francisco, CA: Morgan-Kaufmann, 1999.

[22] Hofmeyr, S., A. Somayaji, and S. Forrest. "Intrusion Detection using Sequences of System Calls." *J. Comp. Security* **6** (1998): 151–180.

[23] Hofmeyr, S. A., and S. Forrest. "Architecture for an Artificial Immune System." *Evol. Comp. J.* in press.

[24] Hofmeyr, S. A. "Immune System Principles and Mechanisms for Computer Security." Ph.D. diss. proposal, Department of Computer Science, University of New Mexico, 1996.

[25] Hofmeyr, Steven A. "An Immunological Model of Distributed Detection and Its Application to Computer Security." Ph.D. diss., University of New Mexico, Albuquerque, NM, 1999.

[26] Hofmeyr, Steven A. "An Interpretative Introduction to the Immune System." This volume.

[27] Holland, J. H. *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: The University of Michigan Press, 1975.

[28] Holland, J. H., K. J. Holyoak, R. E. Nisbett, and P. Thagard. *Induction: Processes of Inference, Learning, and Discovery.* Cambridge, MA: MIT Press, 1986.

[29] Janeway, C. A., and P. Travers. *Immunobiology: The Immune System in Health and Disease*, 3d ed. London: Current Biology Ltd., 1996.

[30] Kanerva, Pentti. *Sparse Distributed Memory.* Cambridge, MA: MIT Press, 1988.

[31] Kepler, T. B., and A. S. Perelson. "Cyclic Re-Entry of Germinal Center B Cells and the Efficiency of Affinity Maturation." *Immunol. Today* **14(8)** (1993): 412–415.

[32] Lippman, R. "Lincoln Laboratory Intrusion Detection Evaluation." October 1999. ⟨http://www.ll.mit.edu/IST/ideval/index.html⟩.

[33] Mukherjee, B., L. T. Heberlein, and K. N. Levitt. "Network Intrusion Detection." *IEEE Network* (1994): 26–41.

[34] Oprea, M., and S. Forrest. "Simulated Evolution of Antibody Gene Libraries under Pathogen Selection." In *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics.* Los Alamitos, CA: IEEE Press, 1998.

[35] Oprea, M., and S. Forrest. "How the Immune System Generates Diversity: Pathogen Space Coverage with Random and Evolved Antibody Libraries." In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 1651–1656. San Francisco, CA: Morgan-Kaufmann, 1999.

[36] Oprea, Mihaela L. "Antibody Repertoires and Pathogen Recognition: The Role of Germline Diversity and Somatic Hypermutation." Ph.D. diss., University of New Mexico, Albuquerque, NM, 1999.

[37] Osmond, D. G. "The Turn-Over of B-cell Populations." *Immunol. Today* **14(1)** (1993): 34–37.

[38] Percus, J., O. Percus, and A. S. Perelson. "Predicting the Size of the Antibody-Combining Region from Consideration of Efficient Self/Non-Self Discrimination." *Proc. Nat. Acad. Sci.* **90** (1993): 1691–1695.

[39] Perelson, A. S., and F. G. Oster. "Theoretical Studies of Clonal Selection: Minimal Antibody Repertoire Size and Reliability of Self-Nonself Discrimination." *J. Theor. Biol.* **81** (1979): 645–670.

[40] Perelson, A., R. Hightower, and S. Forrest. "Evolution (and Learning) of V-Region Genes." *Resh. Immunol.* **147** (1996): 202–208.

[41] Segel, L. A., and R. Lev Bar-Or. "Immunology Viewed as the Study of an Autonomous Decentralized System." In *Artificial Immune Systems and Their Applications*, edited by D. Dasgupta, 65–88. Berlin: Springer-Verlag, 1998.

[42] Segel, L. A., and R. Lev Bar-Or. "On the Role of Feedback in Promoting Conflicting Goals of the Adaptive Immune System." *J. Immunol.* **163** (1999): 1342–1349.

[43] Seidman, J. G., A. Leder, M. H. Edgell, F. Polsky, S. M. Tilghman, D. C. Tiemeier, and P. Leder. "Multiple Related Immunoglobulin Variable Region Genes Identified by Cloning and Sequence Analysis." *Proc. Natl. Acad. Sci. USA* **75** (1978): 3881–3885.

[44] Smith, D. J., S. Forrest, D. H. Ackley, and A. S. Perelson. "Using Lazy Evaluation to Simulate Realistic-Size Repertoires in Models of the Immune System." *Bull. Math. Biol.* **60** (1998): 647–658.

[45] Smith, D. J., S. Forrest, D. H. Ackley, and A. S. Perelson. "Variable Efficacy of Repeated Annual Influenza Vaccination." *Proc. Nat Acad Sci.* **96** (1999): 14001–14006.

[46] Smith, D. J., S. Forrest, R. Hightower, and A. S. Perelson. "Deriving Shape Space Parameters from Immunological Data for a Model of Cross-Reactive Memory." *J. Theor. Biol.* **189** (1997): 141–150.

[47] Smith, Derek. "Towards a Model of Associative Recall in Immunological Memory." Technical Report 94-9, University of New Mexico, Albuquerque, NM, 1994.

[48] Smith, Derek. "The Cross-Reactive Immune Response." Ph.D. diss., University of New Mexico, Albuquerque, NM, 1997.

[49] Smith, Derek, Stephanie Forrest, and Alan Perelson. "Immunological Memory is Associative." In *ICMAS workshop on "Immunity Based Systems,"* 1996.

[50] Smith, R., S. Forrest, and A. S. Perelson. "Searching for Diverse, Cooperative Populations with Genetic Algorithms." *Evol. Comp.* **1(2)** (1993): 127–149.

[51] Somayaji, A., and S. Forrest. "Automated Response Using System-Call Delays." In *Usenix Security Syposium 2000*, submitted.

[52] Tonegawa, S. "Somatic Generation of Antibody Diversity." *Nature* **302** (1983): 575–581.

[53] Warrender, C., S. Forrest, and B. Pearlmutter. "Detecting Intrusions Using System Calls: Alternative Data Models." In *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, 133–145. Los Alamitos, CA: IEEE Computer Society, 1999.

[54] Weigert, M. G., I. M. Cesari, S. J. Yonkovitch, and M. Cohn. "Variability in the Light Chain Sequences of Mouse Antibody." *Nature* **228** (1970): 1045–1047.

[55] Wilson, M., R. Seymour, and B. Henderson. "Bacterial Perturbation of Cytokine Networks." *Infect. & Immunol.* **66** (1998): 2401–2409.