Artificial Immune Systems

Sabine Bachmayer
Departement of Computer Science
Gustaf Haellstroemin katu 2b
Finland, 00014 University of Helsinki
sabine.bachmayer@helsinki.fi

Manuscript received in March 2007

***Abstract*—The study of Artificial Immune Systems (AIS) includes the development of computational abstractions of the natural immune system. The most intuitive application of these methods is the virus and in general the intrusion detection of a single system or network. But with the growing amount of data, global interconnectivity and communication also other fields of application for Artificial Immune Systems came up.**

***Index Terms*—Artificial Immune Network, Artificial Immune Algorithms, Immune Networks**

I. Introduction and Terms of Definition

It is difficult to give a fundamental definition of an Artificial Immune System. One could say it is a model of the natural immune system that can be used by immunologists for explanation, experimentation and prediction activities - also known as computational immunology. Another definition is that an AIS is an abstraction of an immunological process which protect creatures (human beings and animals) from intrusions via foreign substances which might be also a useful idea for the field of computer science and which are presented in this paper. [1]
The topic of Artificial Immune Systems is in a large part oriented on the natural immune system. Because of that the following terms of definition are necessary for reading this paper fluently.

- *Antibody* - A specialized immune protein which is produced because of the intrusion or injection of foreign substances into the body
- *Anomaly / Antigen* - A foreign substance, which stimulates the production of antibodies when it is injected into the body. Antigens can include toxins, bacteria, foreign blood cells, and the cells of transplanted organs.
- *T-cell* - A special type of white blood cell that is of key importance to the immune system. It has so called T-cell-receptors (TCR) on its surface with that it can detect antigenes. Normally the receptors of a T-cell do not match to own substances of the body.
- *B-cell* - A type of white blood cell and, specifically a type of lymphocyte. The B-cell or B-lymphocyte is an immunological important cell.
- *Naive cells* - Mature cells which already have contact to antigens.
- *Proliferation* - Growing and increasing in number rapidly. For example concerning cells it is a rapid cell division and mutation.
- *Crossover* - The biological crossover is the exchange of genetic material between two paired chromosomes. Crossing over is a way to recombine the genetic material so that each person (except for identical twins) is genetically unique.
  In genetic algorithms, crossover is a genetic operator used to vary the programming of chromosomes from one generation to the next. It is an analogy to the biological crossover described above.
- *Genetic Representation* - A way of representing solutions/individuals in evolutionary computation methods. Genetic representation can encode appearance, behavior, physical qualities of individuals.
- *Mutation* - The biological mutation is a permanent change or a structural alteration in the genetic material of humans and many other organisms.
  In genetic algorithms, mutation is a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next. It is analogous to biological mutation described above.
- *MHC* - Major histocompatibility complex. That is a cluster of genes located on chromosome 6 concerned with antigen production.
- *APC* - Antigen-presenting cell. That is a cell which can ”present” antigen in a form that T-cells can recognize it.
- *Medulla* - The innermost part of something.
- *Cortex* - The outer portion of an organ.
- *Thymus* - A lymphoid organ situated in the center of the upper chest just behind the sternum (breastbone). It is in the thymus that lymphocytes mature, multiply, and become T-cells. (That is why they are called T-cells. The T is for thymus.)

If no other reference is given, the definition is taken from Medicine-Net http://www.medicinenet.com/script/main/hp.aspt.

This paper gives an introduction and general overview to Artificial Immune Systems. After this terms of definition, the standard types of AIS, which are Negative Selection, Clonal Selection and Immune Networks are examined. Additionally to these standard types the Genetic Algorithm and the Danger Theory are presented. This is followed by a list of examples and up-to-date science of Artificial Immune Systems to get a better idea of their field of application. Finally an AIS for virus

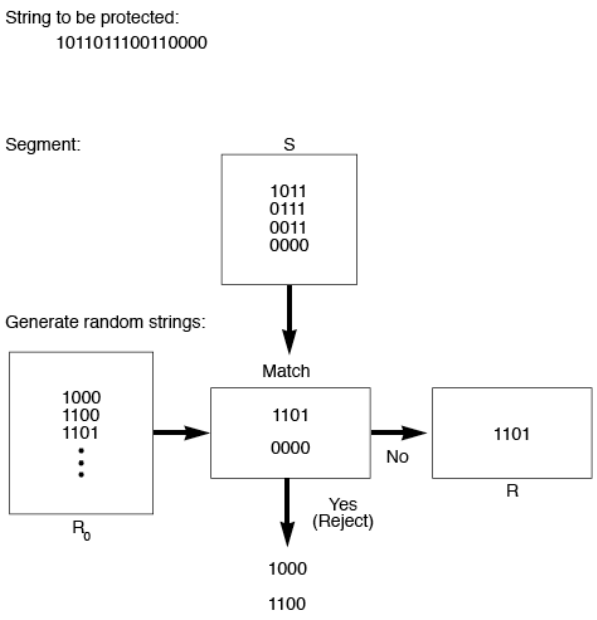

Fig. 1. The generation of the repertoire by using the alphabet (0,1) with r = 2 [2]

detection, developed by Stephen A. Hofmeyr and Stephanie Forrest, is presented in more detailed.

II. Artificial Immune Algorithms

This section describes the most common Artificial Immune Algorithms more detailed.

A. Negative Selection

Forrest et al present in the paper [2] an algorithm which is based on the biological negative selection principle. The algorithm is developed to detect anomalies in a set of strings which could be changed in the checksum, length and so on, done by mal programs like a virus. In "the real world" these strings would be an application program, some data or any other part of a computer system stored in memory.

First the detectors have to be generated which is shown in Figure 1. This happens in a procedure called *censoring*, by splitting the protected string into substrings which produces the collection S of self (sub)strings. In the next step a collection R_0 of random strings is generated. Those randomly generated strings which match self are eliminated, those which do not match any strings of S become a member of the detection collection R - also called *repertoire*. After the repertoire is produced, the monitoring phase can be started, by continually matching strings from S against those from R. In this paper the strings are chosen in the order they were produced for the match.

A simple example would be the self string 0011. If one bit is changed one get for example the string 0111. Then at some point in the monitoring process, it will be recognized, that the changed self string 0111 (from S) matches the detector string 0111 from R.

a) Matching phase: The example above shows a perfect match which means that the two strings are equal in 100%. Of course this was a quite simple example which would never happen in practice - so also perfect matches are quite rarely. For that a matching rule is necessary. In the paper a rule based on nearest neighbor classification is presented which looks for r contiguous matches between symbols in corresponding positions. So two strings s1 and s2 matches if a match at least at r contiguous locations is given.

For example:

String s1: GEWPPVdSSuMQ

String s2: ZPSMXQYSSuGF

They also calculate the probability P_m, that two random strings match at at least r contiguous locations, with the following formula:

$$P_m \approx m^{-r}[(l-r)(m-1)/m+1] \quad (1)$$

where

m = the number of the symbols in the used alphabet,

l = the length of the string,

r = the number of contiguous matches required for a match.

The approximation is only acceptable if $m^{-r} << 1$. On this given formula one can see, that the probability that two random strings will match decreases rapidly with an increasing number of symbols in the alphabet.

B. Clonal Selection

The clonal selection algorithm is modeled on the natural B-cell mechanism. Naive B-cells circulate in the blood and the lymphatic organs. Once the receptors of such a B-cell match to an antigen they proliferate quickly and they also change in order to achieve a better matching. Those B-cells with better matching proliferate again, and so on - so the best matching B-cells are produced. This process is shown in Figure 2. De Castro presents in his paper [3] an algorithm, called CLONALG, which is based on this natural clonal selection where the maintenance of a specific memory set, selection and cloning of the most stimulated antibodies, death of non-stimulated antibodies, affinity maturation and re-selection of the clones proportionally to their antigenic affinity, generation and maintenance of diversity are taken into account.

The following list contains the notation (simplified - for a more detailed view see [3]) which is used later to describe the algorithm.

- $\mathbf{Ab}$: available antibody repertoire;
- $\mathbf{Ab}_{\{\mathbf{m}\}}$: memory antibody repertoire;
- $\mathbf{Ab}_{\{\mathbf{r}\}}$: remaining antibody repertoire;
- $\mathbf{Ag}_{\{\mathbf{M}\}}$: population of antigens to be recognized;
- $\mathbf{f_i}$: vector containing the affinity of all antibodies with relation to the antigen $\mathbf{Ag_j}$;
- $\mathbf{Ab}^{\mathbf{j}}_{\{\mathbf{n}\}}$: n antibodies from $\mathbf{Ab}$ with the highest affinities to $\mathbf{Ag_j}$;
- $\mathbf{C^j}$: population of clones generated from $\mathbf{Ab}^{j}_{\{n\}}$;
- $\mathbf{C^{j*}}$: population $\mathbf{C^j}$ after the affinity maturation process;
- $\mathbf{Ab}_{\{\mathbf{d}\}}$: set of d new molecules that will replace d low-affinity antibodies from $\mathbf{Ab}_{\{r\}}$;

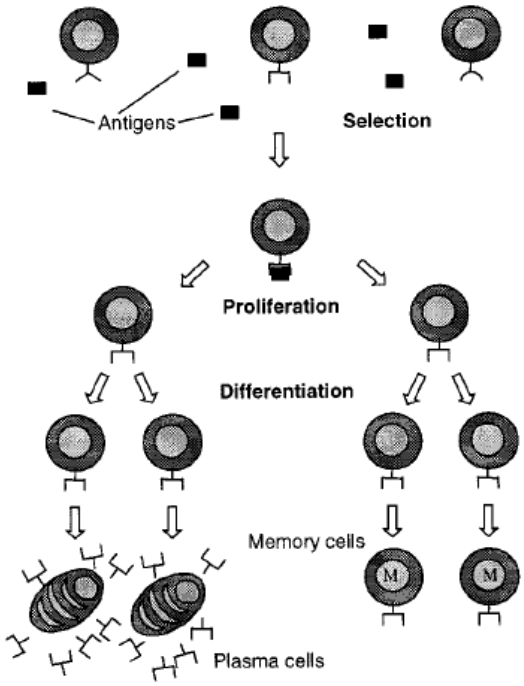

Fig. 2. Clonal Selection Process

- $\mathbf{Ab_j^*}$: candidate, from $\mathbf{C}^{j*}$, to enter the pool of memory antibodies;

Using the termination above, the CLONALG algorithm can be described as followed (see Figure 3) [3]:

1) Choose an antigen randomly from $\mathbf{Ag}_{\{\mathbf{M}\}}$ and present it to all antibodies in the repertoire $\mathbf{Ab}$;
2) Determine the vector $\mathbf{f_j}$ which contains the affinity of the chosen antigen to all the antibodies in $\mathbf{Ab}$;
3) The antibodies with the highest affinity to the chosen antigen are selected from $\mathbf{Ab}$, to compose a new set $\mathbf{Ab}^{\mathbf{j}}_{\{\mathbf{n}\}}$ of high affinity antibodies;
4) These selected antibodies are now cloned independently and proportionally to their affinities to generate another repertoire $\mathbf{C^j}$ of clones. The higher their affinity, the more clones are produced;
5) The repertoire $\mathbf{C^j}$ is submitted to an affinity maturation process inversely proportional to the antigenic affinity to generate another repertoire $\mathbf{C^{j*}}$ of clones. But here is the rule, the higher the affinity, the smaller the maturation rate;
6) Determine the vector $\mathbf{f_j^*}$ which contains the affinity of the matured clones $\mathbf{C^{j*}}$ in relation to the antigen (which was chosen in 1);
7) From $\mathbf{C^{j*}}$ another re-selection is done, to select the one with the highest affinity in relation to the antigen (which was chosen in 1) to be a candidate to enter the set of memory antibodies $\mathbf{Ab}_{\{\mathbf{m}\}}$. If there already exists an antibody (to the antigen chosen in 1) in $\mathbf{Ab}_{\{\mathbf{m}\}}$ which affinity is lower, than it is replaced by the new one;
8) The d lowest affinity antibodies (corresponding to the antigen chosen in 1) from $\mathbf{Ab}_{\{\mathbf{r}\}}$ are replaced by new individuals.

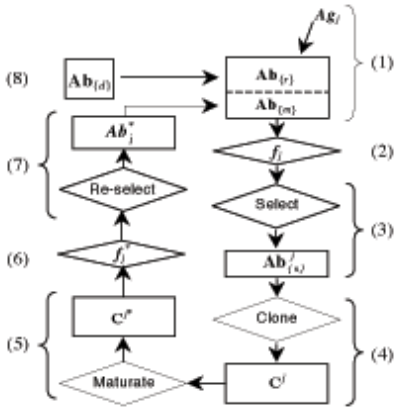

Fig. 3. Computational procedure for the CLONALG algorithm in the pattern recognition phase [3]

The clonal process is also often used for learning algorithms because it is an intrinsic scheme of reinforcement learning strategy where the environment gives rise to the continuous improvement of the system capability.

C. Immune Genetic Algorithm

This algorithm is based on the genetic algorithm which is a search technique, used in computing to find true or approximate solutions, to optimization and search problems. They are categorized as global search heuristics. Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (chromosomes) of candidate solutions (individuals or creatures) to an optimization problem evolves toward better solutions. The typical Genetic Algorithm requires first a genetic representation of the solution domain and second a fitness function for evaluating the solution domain. The standard representation of the solution is a bit-array. The fitness function is always problem dependent and is defined over the genetic representation and measures the quality of the represented solution. Once we have the genetic representation and the fitness function defined, Genetic Algorithm proceeds to initialize a population of solutions randomly, then improve it through repetitive applications of mutation, crossover, and selection operators. The problem with this algorithm is that it has a good ability of global search but it is not very good in local search where the chromosomes decreases quickly [4].

So Chun et al present in another paper [5] a new immune genetic algorithm, which is based on the genetic algorithm, to deal with this problem. In this algorithm the antigen and the antibody are the objective function and the solution and the affinity between an antigen and an antibody is the solution fitness. This method improves the selection operator of the genetic algorithm so it can on the one hand maintain the diversity of chromosomes and on the other hand the performance of global search.

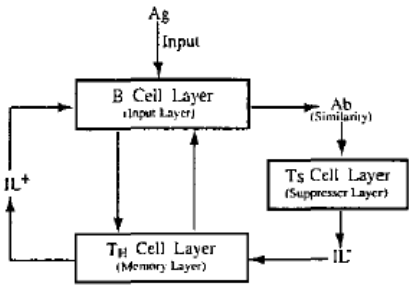

Fig. 4. The Artificial Immune Network designed by Castro et al [7]

D. Immune Algorithm Based on Immune Network

An artificial immune network is a bio-inspired computational model that uses ideas and concepts from the immune network theory mainly the interaction among B-cells and the cloning process. An artificial immune network receives as input antigens and returns an immune network composed of a set of B-cells and connections between them.

The immune network theory was proposed by Jerne [6] as a way to explain the memory and learning capabilities exhibited by the immune system. The principal hypothesis of this theory states that immune memory is maintained by B-cells interacting with each other, even in the absence of foreign antigens. These interactions can be either excitatory or inhibitory. The production of a given antibody stimulates the production of other antibodies and so on. Antigens denotes those molecules that the immune cells are able to recognize and it is necessary to differ between self antigens (antibodies) and non-self antigens. Accordingly with the notation suggested by Jerne the portion on the antigen's surface that an antibody recognizes is named epitope, the portion used by an antibody to recognize antigens is named paratope and the epitope of an antibody is named idiotope. Based on Jerne's work some models of immune network were developed using differential equations to predict the antibody concentration during and after an immune response.

Sun et al describe in [7] an artificial immune network which is based on the natural immune response mechanism described above. For their algorithm they restricted to the interaction between B-cells and T-cells. The participated elements are B-cells, T-helper cells, suppressor T-cells, antigens and antibodies.

The following steps describe the algorithm to Figure 4.

1) The input of this model is a continuous sequence of antigens ($\mathbf{Ag}$) which arrives at the B-cell layer. In this realization the antigens can be similarities between an input pattern and the memory pattern. The $\mathbf{Ag}$ is in form of an N-dimensional vector including the antigens $(A_{g1}, A_{g2}, ..., A_{gn})$ where the value of each $\mathbf{A_{gi}}$ is between 0.0 and 1.0. On the B-cell layer, those B-cells which activity is large enough recognize this input and send out an excitatory signal along a specified path to the T_H-cell layer. This layer is represented by another i x j weight vector $\overrightarrow{W_j}$ where for all i and j an initial condition to $\overrightarrow{W_j}$ is given.
2) When a signal is recognized on a path from the B-cell to the T_H-cell layer, it is multiplied by the pathway's trace $\overrightarrow{W_j}$ as described as followed. First, the minimum value between the vector $\overrightarrow{Ag}$ and $\overrightarrow{W_j}$ is estimated as $min(\overrightarrow{Ag}, \overrightarrow{W_j})$. This makes it possible to correspond to continuous valued input antigens. Then the norm of this minimum value is estimated as $|min(\overrightarrow{Ag}, \overrightarrow{W_j})|$.
3) Now the T_H-cell with the largest stimulus (the largest signals) will be chosen. This happens just by competition interaction in the T_H-cell layer.
4) This phase is called the matching process. The winner T_H-cell sends back another signal to the B-cells - only the matching B-cell reacts and will synthesize the antigen and secretes the antibodies ($\mathbf{Ab}$). These antibodies become input to the T_s-cell layer where the sum of the antibodies is computed and compared to a vigilance parameter r.

It is necessary to train this system, by presenting a set of input patterns to the input of the network, for the adjustment of the weights in a way that similar vectors activate the same T_H-cell. If the same antigens invade again, it is possible to produce an immune response very quickly and producing a large number of antibodies in the network.

Another interesting approach to immune networks is presented in [8] by Vargas et al. They developed an artificial immune network for an autonomous navigation of a robot where the robot corresponds to an organism and every incoming information collected by its sensors (the robot has sensors of color and distance) plays the role of an antigen. The antigens are related to the actual state of the robot (power, direction and so on - see Table I). After the antigens entered the robot, they are recognized by the appropriate antibodies. According to the configuration of the immune network, the concentration of an antibody may stimulate or suppress other antibodies (nodes) of the network. This level of concentration is not primary dependent on the affinity to the antigen but more on the amount of connections (via network links) to other antibodies in the network. Using a roulette-wheel algorithm, an antibody is selected, where antibodies with a higher concentration will have better chances to get selected than others. Table I shows possible antigens and Table II the corresponding actions which are part of the antibodies - the above described dynamic of the network will then decide which action(s) has to be taken. For example for the antigen "internal energy - low" one matching action / antibody could be "Search for home base".

TABLE I
INFORMATION WHICH IS COLLECTED BY THE SENSORS ⇒ TREATED AS ANTIGENS [8]

Antigen	
Garbage	front, left, right, none, carrying
Obstacle	front, left, right, none
Base	front, left, right, none
	close, far, very far
Internal Energy	high, low

TABLE II
THIS ACTIONS ARE PART OF THE ANTIBODIES [8]

Action
Go forward
Turn left
Turn right
Search for Home Base
Catch Garbage
Explore

E. The Danger Theory

This theory is an emerging Artificial Immune System technique based on an abstraction of Matzinger's danger model [9]. The danger theory is cited here because it is a distinct, fast growing alternative/addition to negative selection. Classical immunology defines that an immune response is triggered when the body encounters something non-self but it is yet not completely clear how the differentiation between self / non-self works. But it is known that the maturation process plays an important role to achieve self-tolerance by eliminating those B- and T-cells which reacts to self and, in addition, a second signal is required to trigger an immune response. Matzinger points out that there must be some discrimination going on between self and non-self because, for example, there is no immune response for foreign bacteria which are in the food or air (for further information see [9] and [10]). Matzinger underlines that the danger theory does not introduce new labels but is a way of escaping the semantic difficulties with self and non-self and on this way we can take care of "non-self but harmless" and "self but harmful". Figure 5 shows how an immune response works in the danger theory. A cell that is in destitution sends out an alarm signal, whereupon antigens in the neighborhood are captured by APC. Then they travel to the local lymph node and present the antigens to the lymphocytes - so the danger signal establishes a danger zone around itself. Thus B-cells, which are within the danger zone, get stimulated to produce antibodies that match the antigens and to traverse the clonal expansion process. Those which do not match or are too far away are not stimulated [11].
An interesting approach is using the danger theory in an artificial immune system for data mining problems such as the movie prediction problem described in the paper of Cayzer and Aickelin [12]. In a data mining system it is not possible to differ between self and non-self because the whole system is "self" but these labels can be replaced by interesting and non-interesting data or similar. If it is possible to estimate if interesting data is located "close" or "near" to other interesting data, ideas from the Danger Theory can be useful again. This closeness must not be a physical closeness, it can also be the file size, the correlation of data or similar entry times into the database. A danger signal can be interpreted as an uncovered piece of information which stimulates those antibodies that match data that is "close" to this piece of information [11].
For another example, how the danger theory is used for Artificial Immune Systems, see section IV.

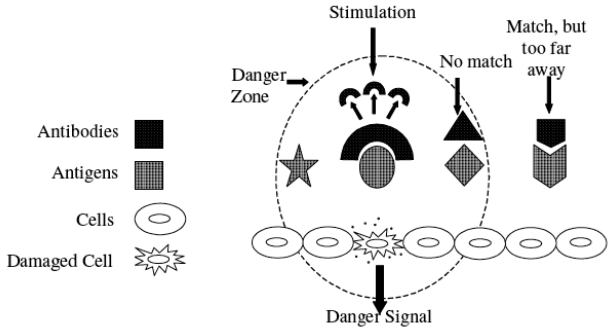

Fig. 5. Model of the Danger Theory [11]

III. APPLICATIONS FOR ARTIFICIAL IMMUNE SYSTEMS

A. Abnormity Detection

Besides Computer Security, Abnormity Detection is one of the most common applications for Artificial Immune Systems. A lot of algorithms are modeled based on the natural immune system. A common part of Abnormity Detection is the virus detection which uses the negative selection algorithm described in II-A.
For example applications please see the next subsection about Computer Security.

B. Computer Security

The most common application area for Artificial Immune Systems is the Computer Security like for example the virus and trojan detection. An example application is presented in [13] by Forrest which uses the Negative Selection Algorithm to detect infected data in a system. In this case the algorithm can recognize, if protected data are changed - like documents, which are infected of a virus. Actually the most common way to detect viruses is with signature strings like the 16-B string which provides a detection rate within 99,5%. The problem here is that these signature strings are hard to create because only some of the $256^{16} = 3.4 * 10^{38}$ combinations identify viruses. And if it would be possible to create one of those strings each microsecond, it would take $1.08 * 10^{25}$ years to create them all. So concerning to the high growth rate of new viruses and trojans there is a need to new detection algorithms. In this field the Negative Selection Algorithm (see II-A) and the Clonal Selection Algorithm (see II-B) are most common. Section IV presents an example system from Stephen A. Hofmeyr and Stephanie Forrest [13] for virus detection with an Artificial Immune System using the Negative Selection in comparison to the Danger Theory.

C. Fault Detection

Fault Detection means to detect malfunctions in a single system or in a network, like a sensor network or client / server network and so on. For that task the Negative Selection principles and the Immune Network Theory are common. Example systems here are the fault detection in sensor networks from Kayama [14] or the learning model from Chen

[15]. In Chen's system the antigens input are classified as self pattern code (the first kind of antigens) and non-self pattern code (the second kind of antigens). The first kind of antigens is used to generate randomly initial antibodies according to the negative selection principle. The second kind of antigens is regarded as learning guide of the immune system [15].

The system of Kayama on the other hand has two execution modes, the training and the diagnosis mode. In the training mode the Learning Vector Quantization (LVQ) extracts a correlation between each two sensors from their outputs. In the diagnosis mode the LVQ contributes to testing each two sensors using the extracted correlation, and the Artificial Immune Network contributes to determining faulty sensors by integrating the local testing results obtained from the LVQ. With the proposed method, faulty sensors, such as age deteriorated ones, which have been difficult to be detected only by checking each sensor output independently, can be specified. [14]
Bradley and Tyrell create a hardware immune system that runs in real-time to monitor continuously a finite state machine for errors. They use the Negative Selection to differentiate between normal and abnormal system operations [16].

D. Optimization and Learning

As already mentioned in II-B, the Clonal Selection Algorithm is a kind of learning algorithm because of the specializing of the B-cells with each infection. So the system learns to become more specialized. For example Chun [17] optimized the exterior shapes of electromagnetism equipments and parameters of synchronous electrical motors based on the Immune Genetic Algorithm (see II-C) and Clonal Selection. De Castro and Zuben present in [18] an approach, using the clonal algorithm, to improve the initial weight vector, which is used in supervised learning for neural networks (see also II-D). It has a strong influence in the learning speed and of course also in the quality of the solution.

E. Data Mining

Data mining is the process of automatically searching large volumes of data for patterns using tools such as classification, association rule mining or clustering. T. Knight and J. Timmis present in [19] an immunological approach to data mining which uses the clonal selection. Their algorithm, called AINE (Artificial Immune Network), is used for pattern discovery and classification in data. AINE uses networks of B-cells which can be represented by an ARB given in the theory of shape space [20]. For more information on AINE take a look to [19] and [21].

This short approach on applications of Artificial Immune Systems shows that the most common algorithms in use are Negative Selection and Clonal Selection, where the Negative Selection is more used in the virus and abnormity detection and the Clonal Selection for learning and optimization problems.

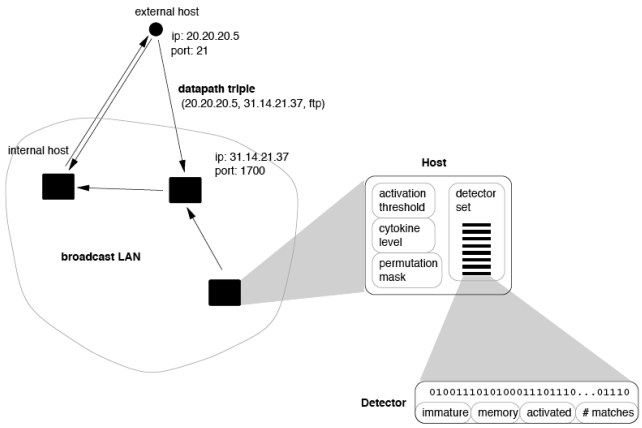

Fig. 6. The system architecture

IV. An Artificial Immune System for Virus Detection

Stephen A. Hofmeyr and Stephanie Forrest present in [13] an Artificial Immune System for protecting a system against foreign attacks. They took the natural immune system as an example for developing their system - so they mapped the natural immunology to computation.

A. Mapping natural immunology to computation

In the natural immune system, many different kinds of cells and molecules exist like lymphocytes, macrophages, natural killer cells, dentritic cells and many others. For this Artificial Immune System the immune system became abstract by using only one kind of basic type of detector cell which combines the most important properties from different cells and which has different states.

Each detector cell consists of a single bit string with the length of 49 bits and, as already mentioned, of different states, which is shown in Figure 6. In this architecture the detection is implemented as a string matching, where each detector is a string d and the detection of a string s occurs when there is a match between s and d according to a matching rule (which could be Hamming distance for example). The used matching rule here is called r-contiguous bits (also see II-A). This rule says that the 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. Where r is a threshold value and determines the specifics of the detector which is an indication of the number of strings covered by a single detector. For example if the r-value equals the length of the detector, it would only match one string, namely itself. So the consequence of a matching rule is that there is a trade-off between the number of detectors used and their specifics. So if the number of specified detectors increases, also the number of detectors required to cover a certain level of detection, increases. The next paragraph describes how these detectors work.

Detector Lifecycle: The detectors are grouped in so called detector sets and within each set new detectors are generated

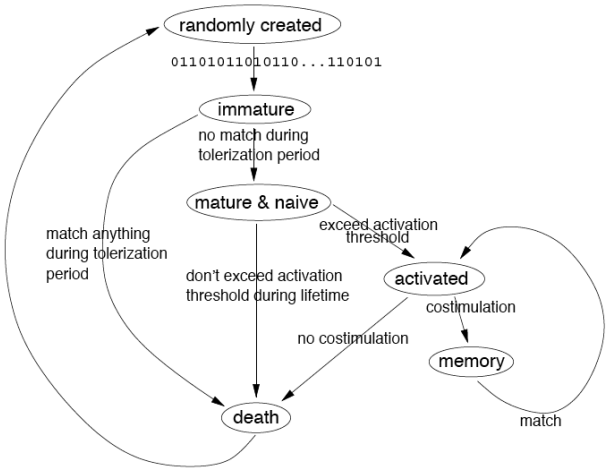

Fig. 7. Detector Lifecycle

randomly asynchronously - similar to the natural system. As figure 7 shows, the new generated detectors have to go through a negative selection (see II-A). Only the detectors which survive the negative selection are then so called ”immature” detectors for a certain period of time called toleration period. After the detector has passed this period, which means if it matches a sufficient number of non-self packets, it becomes an active detector and will exist for a finite lifetime otherwise it will be replaced by a new detector. After this lifetime it becomes a memory detector if it has passed its match threshold. If not it is deleted and replaced. The co-stimulation in the picture means that it enters the competition which is another learning mechanism and described in the next subsection.

B. Learning Mechanisms

Another learning mechanism, besides negative selection and maturation of naive cells into memory cells, is the affinity maturation. In its simplest form, detectors compete against each other for non-self-packets. So if two detectors match simultaneously the same packet, the detector with the closest match wins. In this way, there is more pressure to discriminate more precisely between self and non-self which should avoid auto-immune-reactions. Detectors which are successful will then proliferate themselves which means that they copy themselves and migrate to other computers.
One problem in such an immune system is the possibility of false alarms. For that so called second signals are used. They also exist in the natural immune system in the way that T-helper lymphocytes exist. When a B-lymphocyte binds a foreign peptide (= first signal), a T-helper cell is necessary to trigger an immune response which prevents B-lymphocytes of acting against themselves. In this AIS, the T-helper-cell is a human which has to confirm the alarm by sending an e-mail to the detector within a specific time period (for example 24 hours). If no mail (the second signal) arrives, the AIS assumes it was a false alarm and destroys the detector.

C. Results

The AIS described above was implemented and tested on a sub-net of the University of Mexico which consists of 50 machines on a switched segment. The results were achieved with 100 detectors per host with a match length of 12 and 49-bit detectors [13]. *Test set*: A self and a non-self data set were used for the test. The self set consists of normal traffic which was collected during the test period of 50 days. The non-self set consists of traffic generated during intrusive activity. Without using the co-stimulation and threshold activation, the false-positive rate was 94%. After using the threshold activation the false-positive rate dropped to about 8 % and after using the co-stimulation it dropped to about 5%. Each of these false alarms consisted of a small set of anomalous packets. So this rate is quite good compared to the state-of-the-art systems in use. The non-self-set consists of 8 intrusive incidents with 7 faithful logs of real incidents and one incident were simulated attacks from different locations. The result of this test was that 7 of the 8 incidents were detected correct.

D. Another approach using the Danger Theory

The problem with the system described above is that an immune response requires an infection beyond a certain threshold but on the other hand in this way the operator might get fewer alarms which is not very useful for an autonomous system. Further more for such a system one has to define self and non-self which is not always clear.
Using the danger theory can help to solve some of these problems. For example the discrimination between self and non-self might be helpful but is not longer essential because not the non-self but the danger signal causes the immune response now. *Danger Signal*: The danger signal should come up after detecting minimal infection to avoid the system getting damaged. How to realize such a signal certainly depends on the system but the following list shows some examples:

- Too low or too high memory usage
- uncaused disc usage
- unexpected changes of files detected by changes in the checksum or filesize
- SIGABRT signal from abnormally terminated UNIX processes
- presence of non-self (if a differentiation between self and non-self is given)

After sending the danger signal, the immune system can start to react with ”antigens” which could for example be executables or connections which are ”near” (not necessarily in a geographical or physical way) the sender of the signal. Thus only those antibodies that match those antigens within a radius will proliferate. If the dangerous components are then identified, other measures can be taken like sending it to a special part of the system simulating another attack. So there is the advantage of not having to send all detectors to confirm danger.

V. Conclusion

This paper shows that the principles of the Artificial Immune Systems are quite complex and abutted to the natural

immune system. Concerning the natural immune system there are a lot of questions still open caused by its complexity which certainly hinders the development of the Artificial Immune Systems. Compared with, for example neural networks or genetic algorithm, different Artificial Immune System approaches, models and algorithms for different applications were designed during the past 15 years. So there is no standard analytic design guidance yet.

Besides that this field of science offers new opportunities to deal with different problems, for example in searching and security, which are bestowed with the growing technology. This paper immerses not very deep into this material but gives a short overview to the state-of-the-art methods.

REFERENCES

[1] S. M. Garrett, "How do we evaluate artificial immune systems?" *Evol. Comput.*, vol. 13, no. 2, pp. 145–177, 2005.

[2] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri, "Self-nonself discrimination in a computer," in *SP '94: Proceedings of the 1994 IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 1994, pp. 202–212.

[3] L. N. de Castro and F. J. V. Zuben, "Learning and optimization using the clonal selection principle," in *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, June 2002, pp. 239 – 251. [Online]. Available: \url{http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1011539}

[4] L. Wang and L. Jiao, "The immune genetic algorithm and its convergence," in *Fourth International Signal Processing Proceedings*, 1998, pp. 1347 – 1350.

[5] J. Chun, M. Kim, H. Jung, and S. Hong, "Shape optimization of electromagnetic devices using immune algorithm," in *IEEE Transactions on Magnetics*, vol. 33, no. 2, 1997, pp. 1876 – 1879.

[6] J. N.K., "Towards a network theory of the immune system," *Ann. d'immunologie*, vol. 125, pp. 373 – 389, 1974.

[7] S. W., T. H., T. Z., and I. M., "An artificial immune network with diversity and its applications," in *IEEE EmBS Asian-Pacific Conference on Biomedical Engineering*. IEEE Computer Society, 2003, pp. 326 – 327.

[8] V. P.A., de Castro L.N., M. R., and von Zuben F.J., "Implementation of an immuno-genetic network on a real khepera ii robot," in *The 2003 Congress on Evolutionary Computing*. IEEE Computer Society, 2003, pp. 420 – 426.

[9] P. Matzinger, "The danger model: A renewed sense of self," *Science Magazine*, vol. 296, no. 5566, pp. 301–305, 2002.

[10] ——, "The danger model in its historical context," *Scandinavian Journal of Immunology*, vol. 54, pp. 4–9, 2001.

[11] U. Aickelin and S. Cayzer, "The danger theory and its application to artificial immune systems," in *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, 2002, pp. 141–148.

[12] S. Cayzer and U. Aickelin, "A recommender system based on the immune network," in *Poceedings of the 2002 Congress on Evolutionary Computation*, 2002, pp. 807 – 812.

[13] S. A. Hofmeyr and S. Forrest, "Immunity by design: An artificial immune system," in *Proceedings of the Genetic and Evolutionary Computation Conference*, W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, Eds., vol. 2. Orlando, Florida, USA: Morgan Kaufmann, 1999, pp. 1289–1296.

[14] M. Kayama, Y. Suqita, Y. Morooka, and S. Fukuoka, "Distributed diagnosis system combining the immune network and learning vector quantization," in *Industrial Electronics, Control and Instrumentation*, vol. 2, 1995, pp. 1531– 1536. [Online]. Available: \url{http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=10334&arnumber=484178}

[15] Q. Chen and D. Zheng, "A model for detection and diagnosis of fault based on artificial immune theory," in *Mechatronics and Automation*, 2006, pp. 2443 – 2447. [Online]. Available: \url{http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=4026024&arnumber=4026483&count=464&index=456}

[16] D. Bradley and A. Tyrrell, "A hardware immune system for benchmark state machine error detection," in *Proceedings of the 2002 Congress on Evolutionary Computing*, vol. 1, 2002, pp. 813–818.

[17] J. Chun, J. Lim, H. Jung, and J. Yoon, "Optimal design of synchronous motor with parameter correction using immune algorithm," in *IEEE Transactions on Energy Conversion*, vol. 14, no. 3, 1999, pp. 610 – 615.

[18] L. N. de Castro and F. J. V. Zuben, "An immunological approach to initialize feedforward neural network weights," in *Proceedings of the 5th International Conference on Artificial Neural Networks and Genetic Algorithms*, 2001, pp. 126–129.

[19] T. Knight and J. Timmis, "Aine: An immunological approach to data mining," in *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*. Washington, DC, USA: IEEE Computer Society, 2001, pp. 297–304.

[20] D. L. Chao and S. Forrest, "Information immune systems," *Genetic Programming and Evolvable Machines*, vol. 4, no. 4, pp. 311–331, 2003.

[21] J. Timmis and T. Knight, "Artificial immune systems: Using the immune system as inspiration for data mining," in *Data Mining: A Heuristic Approach*, H. A. Abbass, R. A. Sarker, and C. S. Newton, Eds. Group Idea Publishing, September 2001, ch. XI, pp. 209–230. [Online]. Available: \url{http://www.cs.kent.ac.uk/pubs/2001/1221}

All links were validated on 6^{th} of March 2007