Using Positive and Negative Selection from Immunology for Detection of Anomalies in a Self-Protecting Middleware

Andreas Pietzowski, Benjamin Satzger, Wolfgang Trumler, Theo Ungerer
Institute of Computer Science, University of Augsburg
D-86159 Augsburg, Germany
{pietzowski,satzger,trumler,ungerer}@informatik.uni-augsburg.de

Abstract: The immune system of our body matured over millions of years and has proven its functionality. In our research we are going to transfer techniques of a biological immune system to a computer based environment. Our goal is to design a self-protecting middleware which isn't vulnerable to malicious messages. First off this paper proposes an artificial immune system with negative and positive selection and evaluates optimal parameter settings. Our tests showed that the detection rate of unknown malicious messages can reach up to 99.6%. Further on we propose techniques to minimize the memory space needed for storing the antibodies and to speedup the time needed for detecting malicious messages. We obtained a space minimization by 30% and gained a speedup of 30 with execution time optimization.

1 Introduction

To secure computer systems, current protection applications have to be aware of signatures from viruses or worms that occurred previously. Due to that restriction they cannot recognize or handle brand new intruders that are not already stored in a database. Computer immunology opens up new ways and methods to recognize new intrusions like our biological immune system does and fits well into the research fields of organic computing [MS04] that explore self-organization techniques to achieve computing systems that are self-configuring, self-optimizing, self-healing, and self-protecting. In our organic ubiquitous middleware research [TBPU05] we investigate self-protection techniques to cope with intentionally or unintentionally malicious peers or services. The biologically inspired technique of computer immunology extracts ideas from our human immune system to develop an artificial counterpart [AT05]. Thus the following approach is a first protection step within a self-protecting system. The goal is to develop an immune system which is permissive to good-natured middleware services and messages but can detect appearing malicious events. This immune system is implemented in our organic middleware to evaluate its self-protection behavior in a real environment.

Our previously published investigations [PSTU06b] applied only negative selection to search for anomalies or intrusive messages and in [PSTU06a] we describe the integration in the organic middleware OCμ. The contribution of this paper is the comparative evaluation of positive and negative selection mechanisms, the combination of both, and also a proposal of storage optimization techniques for the binary receptors to increase memory and performance efficiency.

The next section confronts the biological with the artificial immune system and shows the different preferences and their effectiveness. Subsequently we optimize the system as well for memory requirements as also for execution time minimization. In the end we mention related research and end up with conclusions and future work.

2 Ideas from biological immune system for its artificial counterpart

A look at the functionality of the biological immune system shows that the basic requirement of such a system is to distinguish between harmless objects called *selfs* and harmful objects called *nonselfs* [HF00]. Matching works due to different protein patterns on the surface of the unknown objects and the antibodies. All mammals have a *thymus* which is aware of all selfs known in the body. Antibodies are continuously created with random protein surfaces and get singled out with *negative selection*. If such a cell is activated in the thymus it will be destroyed [HF00] otherwise it will be released to the body to protect it against a specific type of intruder. The opposite process called *positive selection* is also used in the body [SBE04] which focuses a T-cell receptor to recognize peptides bound to molecules of the major histocompatibility complex (MHC). It works similar to the negative selection process but in difference it singles out all protein surfaces which *can* bind to selfs in the thymus. Those receptors which do not bind are destroyed. The biological immune system is extremely distributed besides the centralized work of the thymus.

The working domain in our middleware-based immune system is not proteins but binary messages. The goal in the artificial immune system is also to distinguish between foreign nonself messages and messages which are tolerated in the system because they belong to self. This system should also be distributed to prevent the middleware from ending up unprotected if one central detection node in the system gets disabled or unreachable. To avoid this we abided by the highly distributed biological paradigm. Instead of storing the huge amount of information about all selfs on every node we want to filter out messages with comparison to short bit strings. Thus we first developed antibodies analogous to the r-chunk method [SBE04]. The antibodies are represented by a short bit string of length r which embodies the receptor. Additionally they have a specific offset where they start their comparison to the messages. Before the antibodies get released to the system they are perambulated with the negative selection process. The opponent of the antibodies is the group of so-called probodies[1]. They are designed the same way as the antibodies but in difference are generated with the use of the positive selection mechanism.

Instead of waiting for all newly created antibodies and probodies until they perambulated the human thymus we preferred to generate them in a structured way (see 3.2). This works because the middleware is aware of all its self messages and thus it can create all receptor patterns not used by any self message at a specific offset.

[1] We coined the word probody because there is no handy notion in immunology for the item that is required in our artificial immune system.

3 The design of the artificial immune system

With the approach described in the previous section we have different values which have to be considered or adjusted to optimize the detection of nonselfs in a decentralized system: (1) the length of the self messages, (2) the structured generation of the receptors, (3) the amount of different offsets or groups, and (4) the amount of antibodies and probodies distributed in the system. To evaluate the immune system approach we implemented the artificial immune system into our middleware OCμ and we separated the key components from the system and built a simulation environment for faster evaluations.

3.1 The length of the messages

Instead of fiddling around with messages of different and unknown length we transformed all the messages appearing in the system to a unified length by using a hash algorithm. Now the optimal length of the resulting hash value has to be determined and the most appropriate hash algorithm has to be chosen.

In our system we have s self messages. Antibodies and probodies always compare themself to the hashed incoming messages at r contiguous bits starting at a specific offset o. We regard different sections within the set of all hashed self messages, where a section is defined as the part a receptor should compare to. The sections of all hashed self messages with the same offset and same length form a group (see figure 1). Every group consists of s subsets of the hashed messages and every subset has a length of r bits. The usage u of a group is defined as the percentage of different bit patterns in that group compared to all possible bit patterns of length r. u is essential for the amount of receptors which can be created in that group. The amount of possible antibodies at a given offset is $(1 - u) \cdot 2^r$ whereas the amount of possible probodies is $u \cdot 2^r$. It is evident that a shorter length of the receptors results in fewer possible antibodies or probodies and a longer receptor would result in a bigger amount which can cover more nonselfs or selfs. The latter also uses more space to distribute them in the system which may result in more overhead (see section 4.2 for optimizations in that aspect).

We needed an algorithm that produces messages of fixed length and chose the hashing algorithm of MD5 which produces strings of 128 bits in length. Naturally duplicates are inevitable due to the usage of a hash function but this is necessary to reach a fixed length for all messages. In case the MD5 algorithm produces too many duplicates another hash algorithm with more than 128 bits should be chosen. That issue will not be discussed any further in this paper because it is only a parameter for scalability.

3.2 The structured generation of the receptors

Because the system is aware of all the harmless self messages it is able to analyze the binary distribution within the different groups. Our algorithm produces all possible an-

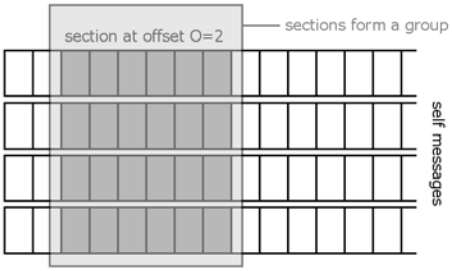

Figure 1: Example how a section of selfs builds a group

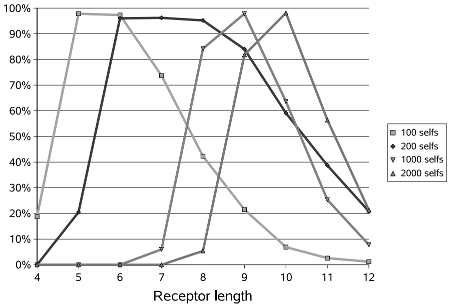

Figure 2: Recognition probability of antibodies

tibodies and probodies by starting with the smallest receptor length of one bit. It subsequently checks all possible bit patterns with this receptor length at every offset if they are suitable for either an antibody or a probody. This is repeated while the receptor length increases. Because there is no upper bound for the receptor length the loop can be exited when enough receptors were created (e.g. when all receptors with the optimal receptor length were created).

For determining the optimal length of the receptors we set up different simulator configurations with different amounts of self messages and tested the detection ability of antibodies with different receptor lengths. The detection rate of the different types of antibodies is shown in figure 2 where always the maximum amount of antibodies were created for every receptor length. This experiment demonstrates that it is not efficient to choose receptors with a fixed or even a random length. Instead it shows that an optimal value has always to be adjusted due to the amount of self messages known in the system. With mathematical calculations it can be shown that the best detection can be reached when a receptor length of about $log_2 s$ is used [PSTU06b]. This knowledge is helpful to generate as few receptors as possible but with a maximum detection rate.

3.3 The amount of offsets

In our research we tested different kinds of offsets within the messages used to match the antibodies with a hashed message to detect malicious objects. The easiest way is to define only a single offset and generate antibodies with receptors of length r which compare to the message only at this specific offset. But this is exactly the same as if the hashed messages would have the same size as the receptors and the whole incoming messages always compare to the antibody. This would result in a very bad detection of nonselfs as the usage of this group corresponds to the probability for not recognizing the nonselfs.

In contrast in another test environment we designed the antibodies to use all possible offsets. This results in fully overlapping groups and thus less messages can be generated that don't match any antibody. As expected our tests showed that the more we over-

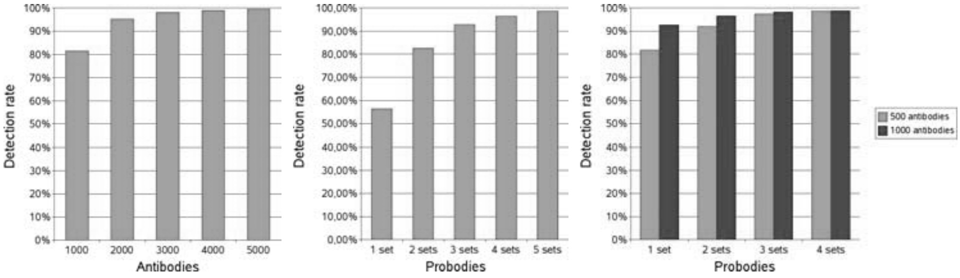

Figure 3: Mean detection rate of nonselfs when using different amounts of antibodies, different amounts of sets of probodies, and a combination of antibodies and probodies

lapped the groups the better the detection of nonself messages worked out when always the same amount of antibodies was created. When using all possible offsets (from offset 0 to $messagelength - r$) the best detection was observed. In difference we do not use every possible offset for the probodies but only offsets with $o \, mod \, r = 0$. Other offsets would result in unintentionally redundant and unnecessary parts of probodies.

3.4 The amount of antibodies and probodies distributed in the system

Indeed the more antibodies and probodies are located at a node in the system the more we achieve an optimal detection rate. Our goal is not to detect all intrusive messages but a suitable amount with as few time and space effort as possible. Due to this aspect we set up simulation environments to test the detection rates with different amounts of antibodies and probodies. The simulation environment consists of 1000 self messages and we always ran different tests to achieve average values which are shown in figure 3. The first chart shows the amount of negative selected antibodies and their detection rate. When using 5000 antibodies – with a receptor length of only nine bits compared to the length of the hashed self messages – we reached a detection rate of 99.6%. Comparing messages to antibodies differs from comparing to probodies. As soon as one antibody matches a pattern, the message is treated as nonself. The probodies are always generated in sets. A set of probodies consists of probodies that cover all self messages at a specific offset and receptor length. In other words, one probody from each self message will be chosen and put into the set. To identify a self at least one probody out of every probody group has to match the message. In our test those sets contained about 250 probodies. When comparing a message to the probodies it is treated as self if every available set of probodies matches. In the second chart in figure 3 the detection rate reaches up to 98.6% when comparing to five probody sets. The problem about the comparison to probodies is that always every probody from every set has to be compared to the message which takes time for computation. A combination of both detection mechanisms result in an improved detection rate and simultaneously less receptors to be stored. These results were shown in the last chart in figure 3.

A small example also demonstrates the benefit of using antibodies in combination with probodies when in sum always an amount of 1250 receptors is used. The results in table 1 show that the detection rate is better when only antibodies are used compared to the exclusive use of probodies. However the best results were gained when using 1000 antibodies and one set of probodies. This is explainable because antibodies are spread to every offsets whereas five sets of probodies can only compare at five different offsets. An increase of antibodies or probodies always improves the detection rate but this test should demonstrate the different power of both techniques. Although the usage of probodies improves the detection rate there is always also a little drawback in execution time because every probody has to be checked for matching and thus has to be weighted up depending on the system. However, in the normal test case of non-malicious items all antibodies have to be checked, too, only in the case of matching an antibody to the pattern of a malicious item the test could be aborted and execution time saved.

amount of antibodies	1250	1000	750	500	250	0
sets of probodies	0	1	2	3	4	5
detection rate	87.03%	94.12%	93,90%	91,63%	84,26%	79,87%

Table 1: Performance of the detection rate when using both antibodies and probodies but always a sum of about 1250 receptors.

4 Optimizations

4.1 Optimizing memory requirements by merging receptors

We propose a technique for memory requirement minimization which doesn't advance the detection probability but minimizes the amount of receptors to be stored in the system. Two receptors with the same offset can be replaced by one receptor with a wildcard if they differ at only one position on the receptors. Also receptors with wildcards can be merged together when two receptors have the same wildcard positions and differ at exactly one further position.

In our tests we were always able to eliminate about 30% of the receptors by merging them together. Unfortunately in our test cases we never were able to produce receptors with three or more wildcards because this is very difficult when the group usage exceeds some specific value.

4.2 Optimizing execution time

When storing n receptors of length r in an array the comparison against an incoming message would result in a time complexity of $O(n \cdot r)$. Because many receptors correspond to other receptors in specific parts of their pattern, we decided to store them in a binary tree. The comparison starts at the root of the tree and follows the way down according to the binary pattern. If there is a complete way down to the bottom of the tree one receptor stored in that tree matched the compared pattern. For each offset a perfect binary tree of depth r has to be created (without a root node because a receptor can start with both boolean values).

Because every tree has to be perfect and every node consists of one boolean value a memory usage of $(2^{r+1} - 2)$ bits is necessary for all possible o offsets. If an antibody tree matches at one specific path the message is treated as malicious and if all probody trees match the message is treated as good-natured. This comparison results in a time complexity of $O(o \cdot r)$ and because in general $o \ll n$ this approach is much faster than comparing all antibodies with a complexity of $O(n \cdot r)$. Further more o and r are fixed for a group of trees which results in $O(1)$ for comparing incoming messages to all stored antibodies. In our tests the tree comparison resulted in a 30 times faster comparison than using the linear comparison algorithm.

5 Related work

Non-immunological detection systems are mostly signature based or at least they use the information gained from previously detected anomalies. For example current virus scanners contain signatures from known viruses and scan files for a match with those signatures. But there are also approaches which can detect unknown or new intrusions because of not comparing to known patterns but the other way round – using the negative or positive selection [SBE04, HD05].

Hofmeyr and Forrest developed an architecture for an artificial immune system to detect intrusions in a network [HF00]. They also modeled their immune system on the biological immune system described in section 2. The antibodies and the messages in their system consist of bit strings with a fixed length and the system compares the messages appearing in the network to all created antibodies. If r contiguous bits are identical the message is treated as nonself otherwise the message is accepted by the immune system. The number r has to be adjusted in their system to tune the detection probability and can have values from one bit to the entire length of the messages. Our approach follows a similar way to design the antibodies but shows up the basic correlation between the amount of self-messages and the length of the receptors needed to get an optimal detection probability in any domain.

At the Deakin University of Australia they also investigated positive and negative selection processes for anomaly detection. They used co-evolutionary genetic algorithms for learning the patterns and used negative selection to generate synthetic samples of the anomaly class. They simulated two tests where the first one contained no anomalies and the second possessed a few anomalous examples. Their evaluations showed that feeding the system with synthetic anomalies increased the detection rate [HD05]. In contrast we are generating the receptors in a structured way and thus we don't need an evolutionary algorithm to learn the self patterns or generate the synthetic nonselfs.

6 Conclusions and future work

The tests showed that the detection rate was very good (up to 99.6% in some test cases) when choosing a suitable receptor length and an appropriate amount of antibodies. When adding some sets of probodies the detection rate still reached a high detection rate but with far less receptors to be stored. We also looked closer at the speed of detecting selfs and non-selfs and found a way to speed up the comparison enormously by arranging all receptors in a binary tree.

The Organic Computing Middleware for Ubiquitous Environments OCμ [TBPU05] is a message oriented middleware based on a peer-to-peer system. As for now we implemented the generation and distribution of the antibodies and probodies in the middleware environment. The next step will be an automated adaption of the antibodies and probodies if the self set changes in the middleware due to changes in the configuration.

References

[AT05] Paul S. Andrews and Jon Timmis. Inspiration for the Next Generation of Artificial Immune Systems. In *4th International Conference on Artificial Immune Systems*, volume LNCS, pages 126–138. Springer-Verlag, 2005.

[HD05] Xiaoshu Hang and Honghua Dai. Applying both Positive and Negative Selection to Supervised Learning for Anomaly Detection. In *Genetic and Evolutionary Computation Conference*, pages 345–352, Washington D.C., USA, June 2005.

[HF00] Steven A. Hofmeyr and Stephanie Forrest. Architecture for an Artificial Immune System. In *Evolutionary Computation 8*, number 4, pages 45–68, Massachusetts Institute of Technology, 2000.

[MS04] Christian Müller-Schloer. Organic Computing Initiative. http://www.informatik.uni-augsburg.de/lehrstuehle/sik/research/organiccomputing/download/OC-english.pdf, April 2004.

[PSTU06a] Andreas Pietzowski, Benjamin Satzger, Wolfgang Trumler, and Theo Ungerer. A Bio-Inspired Approach for Self-Protecting an Organic Middleware with Artificial Antibodies. In *International Workshop on Self-Organizing Systems*, September 2006.

[PSTU06b] Andreas Pietzowski, Benjamin Satzger, Wolfgang Trumler, and Theo Ungerer. A Practical Computer Immunology Approach for Self-Protection Enhanced by Optimization Techniques. *Journal of Autonomic and Trusted Computing*, September 2006.

[SBE04] Thomas Stibor, Kpatscha M. Bayarou, and Claudia Eckert. An Investigation of R-Chunk Detector Generation an Higher Alphabets. In *Genetic and Evolutionary Computation Conference*, pages 299–307. Springer-Verlag, 2004.

[TBPU05] Wolfgang Trumler, Faruk Bagci, Jan Petzold, and Theo Ungerer. AMUN - autonomic middleware for ubiquitous environments applied to the smart doorplate. *Advanced Engineering Informatics*, (19):243–252, April 2005.