

A BOUNDARY-AWARE NEGATIVE SELECTION ALGORITHM

Zhou Ji

The University of Memphis / St. Jude Children's Research Hospital
Memphis, TN 38152, USA
email: zhou.ji@stjude.org

ABSTRACT

Negative selection algorithms generate their detector sets based on the points of self data. In the approach described in this paper, the continuous self region is defined by the collection of self data. This has important differences from the negative selection algorithms that simply take each self point and its vicinity as the self region: when the training self points are used together as a whole, more information is provided than used as individual points; the boundary between self and nonself regions are detected in the algorithm. It also demonstrated that a negative selection algorithm as a unique strategy can obtain certain results that straightforward positive selection cannot. Experiments are carried out using both synthetic data and real world applications. The former was designed to highlight the difference from conventional point-wise interpretation of self data in negative selection algorithms.

KEY WORDS

Artificial Immune Systems (AIS), Negative Selection Algorithms (NSA).

1 Introduction

Artificial Immune Systems (AIS) has been an active research area that assimilates new inspiration from immune system during the past few years [1]. Although some techniques in this area have not shown convincing distinctiveness and effectiveness, many AIS methods succeeded in a variety of applications [2]. Currently, major types of AIS methods include negative selection algorithm (NSA) [3], gene library and clonal selection model, and immune network model. The negative selection algorithm is one of the earliest methods in AIS. It is believed to have distinct process from alternative methods and be able to provide unique results with better quality [4].

A typical negative selection algorithm is applied in two stages. In the training stage, the detector set is generated. In the detection stage, the detector set is used to detect anomaly in the incoming new data. Although there are many variations, a negative selection algorithm is mainly defined by the following components: the data and detector representation, the generation procedure of the detector set, and the matching rule. The outline of the algorithm is illustrated in fig. 1. The matching rule plays a crucial role, which is used both in training stage to eliminate in-

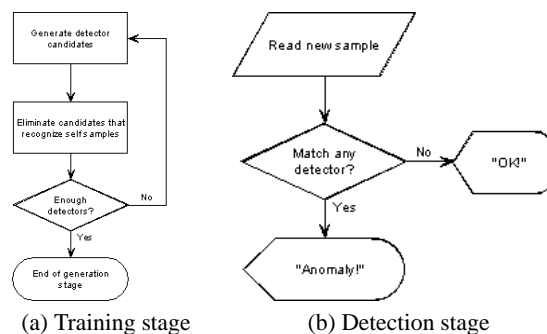


Figure 1. Outline of a negative selection algorithm

valid candidates and in detection stage to report anomaly.

Negative selection algorithms started as a fault detection method. The purpose of this type of methods is to establish a model of normal behavior from the training data - usually a collection of self sample points. Such a model or the knowledge of "self" is usually hard to express in an explicit or concise way. That is part of the reason that soft computing paradigms like AIS become necessary. Depending on the specific representation, it may be a string pattern in string or binary representation, or some regions over the searching space in real-valued representation, or a target concept represented in other forms. In negative selection algorithms, it is implemented as a detector set in the complementary space [3, 5, 6, 7].

Although the negative selection algorithms are in fact a family of algorithms that could be very different from each other, almost all variations interpret the training data in a point-wise way, meaning each self point is used independently to censor the detector candidates and eliminate invalid ones. This paper shows that such a method could lead to bias in the solution, especially around the boundary between self region and nonself region.

Ignorance of this issue could obscure the difference from other aspects of the algorithms. The issue is highlighted by the "boundary dilemma": how much should we generalize the self samples that are close to the boundary between self and nonself regions? Because we don't know which self samples are close to the boundary and which side of them is self region, it is hardly possible to decide the proper interpretation of these samples.

This paper presents a new approach, the boundary-aware NSA, in which the training data are used as a col-

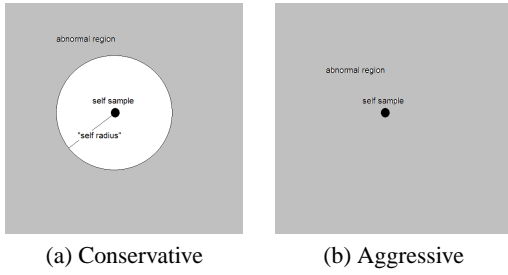


Figure 2. Possible interpretations of a single self sample: Conservative or Aggressive

lection instead of individual points and the existence of boundary are detected. It is based on *V-detector*, which uses variable-sized detectors and terminates training stage when enough coverage is achieved [8]. Using its unique matching mechanism, individual self points cannot claim a region to be self region. Only a group of points together can prevent a region to be covered by detectors.

In NSA, the main mechanism to achieve generalization from self samples is “partial matching”. For example, the most common matching rules in binary representation include *r*-contiguous bits (*rcb*) matching rule, Hamming distance matching rule, and *n*-grams matching rule. In real-valued representation, Euclidean distance matching is usually used. The criterion of these matching rules in essence is based on certain definitions of distance. Although this paper uses Euclidean distance in real-valued representation in the experiments, the same approach can be used in other distance measures or matching rules. We will use a general term “self threshold” to describe either the self radius in real-valued representation or window size *r* in binary representation.

2 Boundary-Aware Negative Selection Algorithm

2.1 Data assumption and issue of boundary

What exactly does a self sample or a collection of self samples mean in term of presenting or defining the self region? This needs to be answered first before we judge any soft computing algorithms of anomaly detection. Regardless of specific algorithm or scheme of negative selection, what a self sample means eventually comes down to the matching rules.

Fig. 2 shows how we may interpret a self sample in different ways. In Fig. 2(a), we assume that a circular area around the self (normal) sample is entirely normal. It is a straightforward way to achieve generalization, similar to partial matching in binary representation in principle. We call an interpretation that allows much variability a “conservative interpretation”, referring to the “conservative” attitude to claim a new sample to be abnormal. In Fig. 2(b),

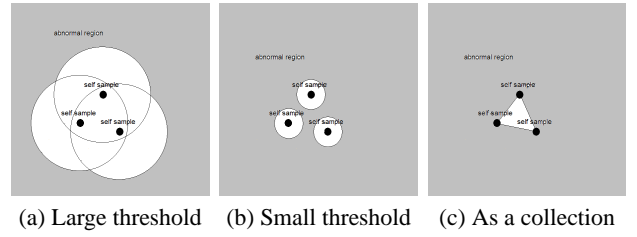


Figure 3. Possible interpretations of a group of self samples

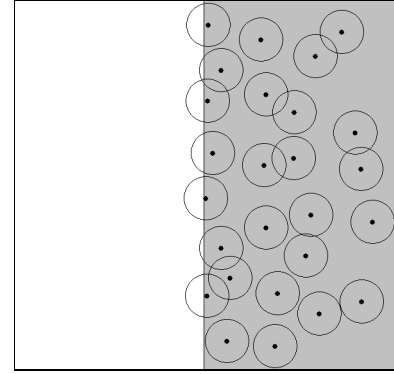


Figure 4. “Boundary Dilemma”

we only consider the exact point of the self sample to be normal. In reality, we may still allow some small deviance because we have to compare float point numbers, but basically we only regard the sample we already see as normal. We call an interpretation that doesn’t allow much variability an “aggressive interpretation”.

At the first look, it seems that in an extremely aggressive interpretation like fig. 2(b), no generalization could happen. That doesn’t have to be the case. Fig. 3 shows a group of three self sample points. Even if we don’t take any circular surrounding area of a single self sample as normal, we can still generalize to a self region by considering the neighboring self points together, as shown in fig. 3(c). Compared with fig. 3(a) or (b), this is more aggressive to detect anomaly, but only to the outside of the perceived “self region”.

Naturally, each self sample point can be interpreted as an evidence that its vicinity is self region. On the other hand, we can fairly assume that the self samples can be drawn anywhere over the entire self region. There is no reason to exclude the points that are close to the boundary between self and nonself regions no matter what kind of matching rule or distance measure is used.

Fig. 4 illustrates the “boundary dilemma”, the scenario that the self samples close to the boundary inevitably extend the actual self region due to the variability allowed by the algorithm. In this figure, the shaded area is the “real” self region; the dots are the self samples and the circles are their generalization. If the self threshold is too small, the space between self samples could not be represented. In

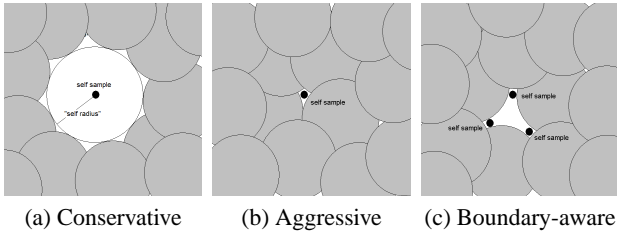


Figure 5. Detectors enclosing the conceived “self region”

other words, more samples are needed to train the system properly. On the other hand, if the self threshold is large, the false self region represented by the boundary samples may be too large to accept.

In the case that the over-covered area is too large compared with the real nonself region, the error would be large. When the nonself region is a thin stripe between two self regions, it may not be able to be represented at all. In those cases, the issue of boundary dilemma will be more considerable.

2.2 Algorithm

The issue described above is tackled by an ingenious simple strategy using a negative selection algorithm to achieve the interpretation illustrated in fig. 3(c).

The above discussion concerns general interpretation of self samples. From the view point of a negative selection algorithm, the difference in interpretation is shown in fig. 5. Fig. 5(a) shows the coverage of a detector set using a conservative interpretation; fig. 5(b) shows one using an extremely aggressive interpretation. Similar to the conceptual discussion, it is possible to generalize the self samples to a finite self region even if detection is extremely aggressive outside the self region. This is illustrated in fig. 5(c).

The key difference between boundary-aware NSA and commonly used point-wise interpretation is shown in fig. 6. \vec{x} is a random point used as the center of detector candidate; r is the detector’s radius to be decided; d is the distance from \vec{x} to the nearest self point; r_{self} is the “self threshold”.

It seems trivial that the difference is only a radius larger by r_{self} , but the effect is more subtle than it appears because the individual self point now does not censor the detector candidates independently. Fig. 5(a) shows one self point and the possible detectors around it when a point-wise algorithm is used. Any detectors cannot get closer to that self point regardless of the distribution of other self points available. When we use a boundary-aware algorithm, however, detectors can be close enough to “touch” the self point (fig. 5(b)(c)), so individual self point cannot prevent any region being covered by detectors. When a collection of self points exist together, however, the detectors cannot be generated within the region as long as the self points are close enough to each other compared with the self thresh-

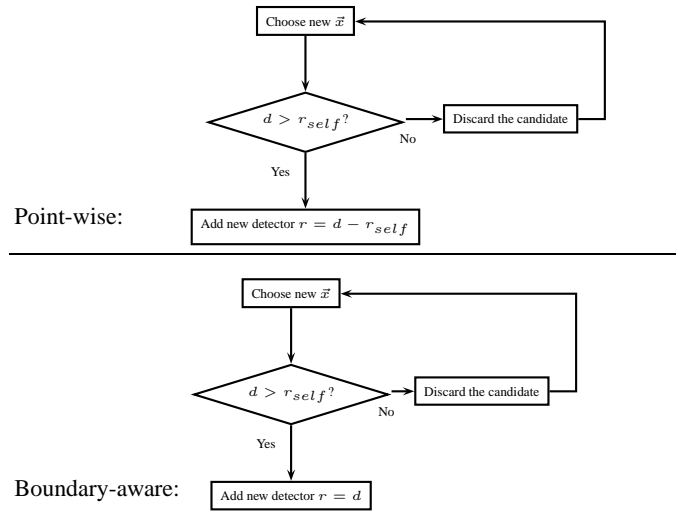


Figure 6. Key difference in boundary-aware NSA

old. That in fact results in a detector set that captures the region as a whole (fig. 5(c)). The boundary of the region is thus detected though not explicitly represented.

It is important to notice that the new strategy cannot be simply translated into different self threshold. The seemingly more aggressive interpretation of a boundary-aware algorithm is not equivalent to a smaller self threshold: it is “aggressive” only in how to deal with the boundary dilemma, not the interior area of assumed self region.

2.3 Applicability

The approach and the discussion are valid for different data representation, distance measure, etc.

The detectors shown in figs. 3 and 5 are illustrated as circles in 2-D real space. That is corresponding to Euclidean distance matching rule in real-valued representation. The same idea perfectly applies to different definition of distance, or different presentation space, e.g. binary representation. For example, for rcb matching rule in binary representation, if window size r is larger, the matching is less likely to occur. That corresponds to a smaller self threshold and more aggressive interpretation. Given r , the boundary-aware algorithm would allow detector to match r bits of any self samples, but not allow to match more than r bits. That will prevent detectors being generated inside the self region. An important difference from the regular rcb matching rule is that r must be variable, which is the strength of V -detector [8].

3 Experiments

Experiments were carried out to verify and demonstrate the strategy presented in the previous section. Since sensitivity and specificity can be balanced using different self threshold, ROC (receiver operating characteristics) curve is used

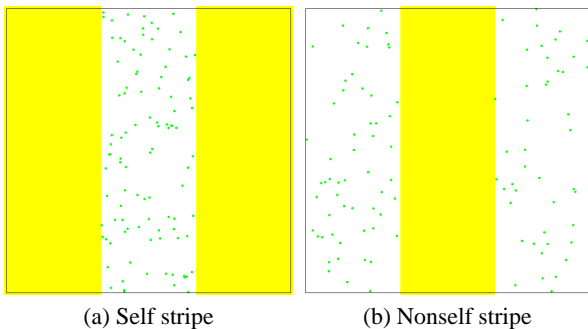


Figure 7. Example of self region in synthetic data

to compare the performance of the algorithms.

3.1 Synthetic data

These experiments were designed mainly to demonstrate the difference between the boundary-aware strategy and the commonly used point-wise interpretation. Real-valued representation and Euclidean distance are used. The entire searching space (universe space) is a 2-D square $[0, 1]^2$. We assume that the training self points are distributed randomly over the self space of certain shape. After the detector set is generated, points randomly distributed over the entire space are used to evaluate the performance.

100 points of self samples are used in most of following results. All the performance measures plotted or tabulated are the average of 100 runs with same control parameters.

3.1.1 Stripes

Because the main issue lies in the boundary between self and nonself regions, a stripe is the simplest shape that can illustrate the difference. Intuitively, the proportion of self area or nonself area that is close to the boundary would affect the results directly, so we tested stripes of different width that are self or nonself. Fig. 7(a) shows the self space that is such a stripe of width 1/3. Fig. 7(b) is a nonself stripe. The white region is self; the shaded region is nonself. The dots over the self region are the training self points.

Fig. 8 is the ROC curves obtained using boundary-aware NSA and point-wise NSA, respectively. Two stripe widths are shown. The boundary-aware algorithm has consistent higher detection rate although its minimum false alarm is higher.

Fig. 9 is the corresponding results for the nonself stripe. The detection rate is lower than self stripe because the area of self region is larger in this case and we are using the same number of self points to train. While the point-wise algorithm can reach wider range of balance, the

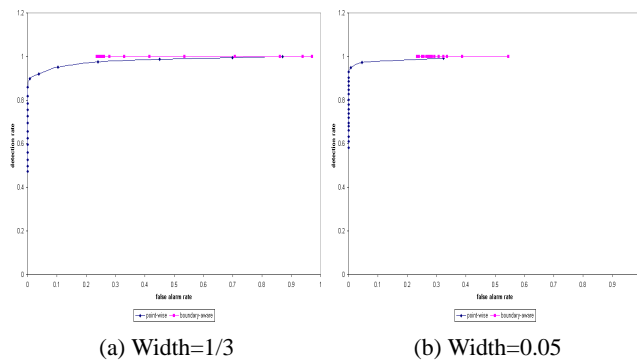


Figure 8. ROC: self stripe

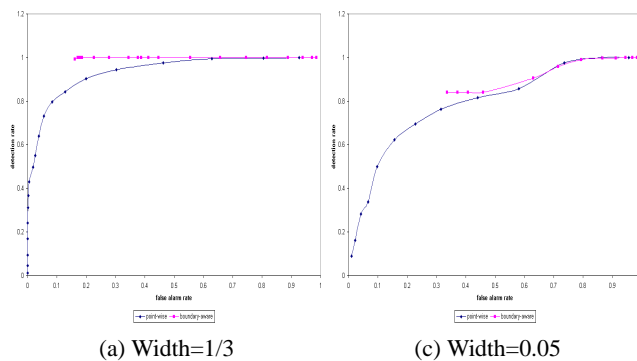


Figure 9. ROC: nonself stripe

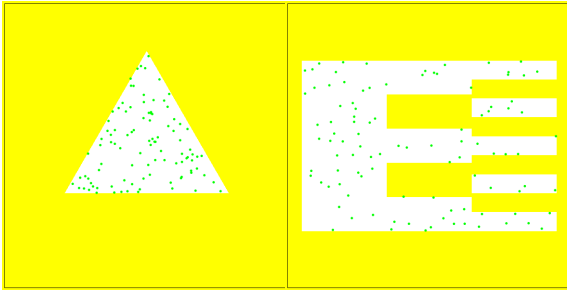
boundary-aware algorithm has better performance generally.

The experimental results confirmed the intuition that when the nonself stripe is very thin, “boundary dilemma” becomes more influential. In other words, the relatively larger portion of nonself region will be identified as self by the point-wise algorithm. The boundary-aware algorithm can improve the detection rate significantly. In this case, because the self samples are not very dense, the increase in false alarm rate by the boundary-aware algorithm is more obvious and offsets its general performance shown on the ROC curve.

3.1.2 Other shapes of self region

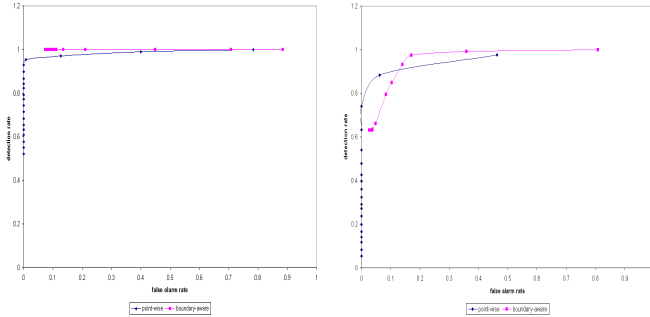
Fig. 10(a) is another simple shape: an equilateral triangle. Fig. 10(b) is a shape with more complicated boundary. It has some stripe type components, especially the right side thin teeth, a large portion of which is close to the boundary. Fig. 11 is the ROC curves for those shapes. The boundary-aware algorithm works better over a wide range of self threshold, but it is not as flexible as the point-wise algorithm to control false alarm.

Figs. 12, 13, 14 illustrate how well the detector set generated from the negative selection algorithm can capture the real shape of the self region. Fig. 12 shows that when



(a) Triangle (b) Comb-shaped region

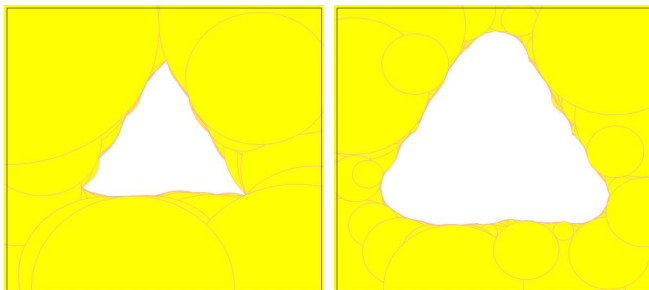
Figure 10. Shapes of the actual self regions



(a) Triangle (b) Comb (1000 training points)

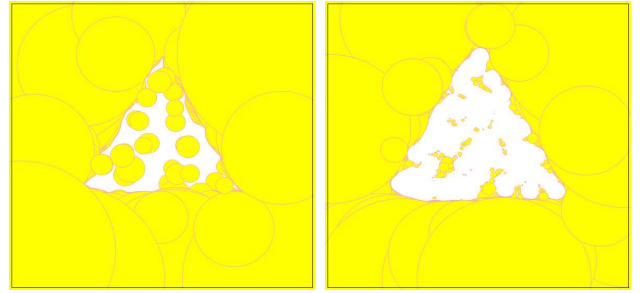
Figure 11. ROC for other shapes

the self threshold is 0.1, relatively large, the detector set generated by the point-wise algorithm was rather insensitive to detect anomaly close to the actual boundary. On the other hand, when the self threshold is smaller, for example 0.03, as in fig. 13, the detector set generated by boundary-aware algorithm was too specific so it raise many false alarms. The boundary-aware algorithm obviously worked to capture sharper edges and pointed angles, showing its capability to detect the boundary. Fig. 14 shows the similar situation for the comb shape when the self threshold is 0.03. For larger self threshold, for example 0.1, neither algorithm can capture the shape faithfully enough to be identifiable.



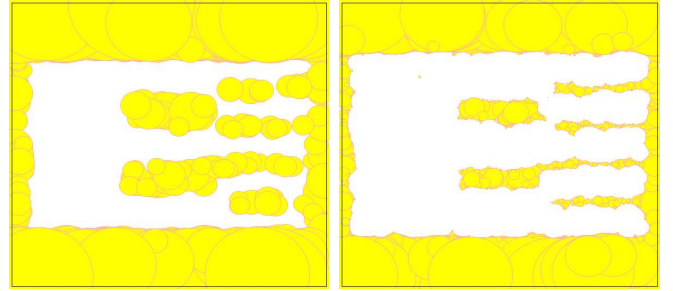
(a) Boundary-aware algorithm (b) Point-wise algorithm

Figure 12. Triangle: Self threshold 0.1



(a) Boundary-aware algorithm (b) Point-wise algorithm

Figure 13. Triangle: Self threshold 0.03



(a) Boundary-aware algorithm (b) Point-wise algorithm

Figure 14. Comb-shaped region: Self threshold 0.03

3.2 Real world applications

3.2.1 Ball Bearing Data

The original data are acceleration signals measured for ball bearings of various conditions [9]. Two different pre-processing methods are used to transform the raw measurements. The first is Discrete Fourier Transform. The other one uses statistical moments up to 5th order. All the available normal data - those from new bearings, are used to train the system. Tables 1 and 2 compare the results obtained using point-wise and boundary-aware algorithms. All these results are average of 100 repeated runs. Self threshold is set as 0.1.

Although the new bearings are obviously normal cases, there is no simple criterion of normality for those bearings that are old but not completely damaged. There-

Table 1. Detected percentage using data pre-processed by DFT

Condition	Number of points	Point-wise	Boundary-aware
New	2739	0%	0.06%
Broken race	2241	94.78%	98.37%
Broken cage	2988	24%	53.03%
Damaged cage	2988	9.21%	26.7%
Badly worn	2988	7.34%	22.42%

Table 2. Detected percentage using data pre-processed by statistical moments

Condition	Number of points	Point-wise	Boundary-aware
New	2651	0%	0.15%
Broken race	2169	74.82%	93.4%
Broken cage	2892	0.52%	6.37%
Damaged cage	2892	0%	2.12%
Badly worn	2892	0%	1.49%

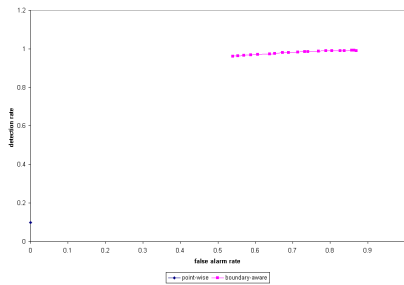


Figure 15. ROC for Indian Telugu data

fore, the actual number of detected cases is reported instead of ROC. We can see better performance of the boundary-aware algorithm.

3.2.2 Indian Telugu Data

This data set consists of 871 patterns of vowel sounds in the language of Indian Telugu. There are six overlapping vowel classes and each pattern is represented by three input features (formant frequencies) [10]. The data is thus 3-dimensional. We chose class 1 as the “normal” class. The first 50 normal samples are used to train the system. Then all the data are used to test the performance. The results in fig. 15 shows that the boundary-aware NSA worked successfully for this application.

4 Conclusion

The boundary-aware negative selection algorithm interprets the training data as a collection instead of as independent points. This algorithm can in fact detect the boundary between the self region and nonself region, though the boundary is not represented explicitly. It obtains certain preferable detection results that positive selection cannot. That supports the claim that negative selection algorithms is a distinctive method from positive selection and other alternatives. We should keep in mind the large variety in negative selection algorithms. Too broad conclusion about this method should not be drawn easily.

The boundary-aware algorithm is meaningful because we assume that the training points are as likely to be close

to the boundary as they are in other parts of the self region. The self threshold is the reference distance when we talk about “being close”. The strength of this approach cannot be substituted by using different self thresholds. The strategy equally applies to different distance measures and data representations.

Relation between the density of training self points and the proper self threshold should be an immediate next goal for formal analysis.

Acknowledgment:

This work was supported in part by NIH Cancer Support Core Grant CA-21765 and the American Lebanese Syrian Associated Charities (ALSAC).

References

- [1] Leandro N. de Castro and Jonathan Timmis. *Artificial Immune System: A New Computational Intelligence Approach*. Springer, 2002.
- [2] Dipankar Dasgupta, Zhou Ji, and F. Gonzalez. Artificial immune system (AIS) research in the last five years. In *Proceedings of The 2003 Congress on Evolutionary Computation (CEC 2003)*, pages 123–130, Canberra, Australia, 9-12 December 2003.
- [3] S. Forrest, A.S. Perelson, L. Allen, R., and Cherukuri. Self-nonsel discrimination in a computer. In *Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy*, Los Alamitos, CA, 1994. IEEE Computer Society Press.
- [4] Simon M. Garrett. How do we evaluate artificial immune systems? *Evolutionary Computation*, 13(2):145–178, 2005.
- [5] Fernando Esponda, Stephanie Forrest, and Paul Helman. A formal framework for positive and negative detection schemes. *IEEE Transactions on System, Man, and Cybernetics*, 2003.
- [6] Dipankar Dasgupta, Senhua Yu, and Nivedita Sumi Majumdar. MILA - multilevel immune learning algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2003)*, LNCS 2723, pages 183–194, Chicago, IL, July 12-16 2003. Springer.
- [7] Fabio A. Gonzalez and Dipankar Dasgupta. Anomaly detection using real-valued negative selection. *Genetic Programming and Evolvable Machines*, 4:383–403, 2003.
- [8] Zhou Ji and Dipankar Dasgupta. Real-valued negative selection algorithm with variable-sized detectors. In *LNCS 3102, Proceedings of GECCO*, pages 287–298, 2004.
- [9] Structural integrity and damage assessment network. World Wide Web, <http://www.brunel.ac.uk/research/cnca/sida/html/data.htm>, 2004.
- [10] S. K. Pal and D. Dutta Majumder. Fuzzy sets and decision making approaches in vowel and speaker recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 7:625–629, 1977.