Migration Threshold Tuning In The Deterministic Dendritic Cell Algorithm

Julie Greensmith[0000−0002−7245−8841]

School of Computer Science, University of Nottingham, UK, NG8 1BB
julie.greensmith@nottingham.ac.uk
https://www.nottingham.ac.uk/computerscience/people/julie.greensmith

Abstract. In this paper we explore the sensitivity of the migration threshold parameter in the Deterministic Dendritic Cell Algorithm (dDCA), one of the four main types of Artificial Immune System. This is with a view to the future construction of a DCA augmented with Deep Learning. Learning mechanisms are absent in the original DCA although tuneable parameters are identified which have the potential to be learned over time. Proposed in this paper is the necessary first step towards placing the dDCA within the context of Deep Learning by understanding the maximum migration threshold parameter. Tuning the maximum migration threshold determines the results of the signal processing within the algorithm, and here we explore a range of values. We use the previously explored Ping Scan Dataset to evaluate the influence of this key parameter. Results indicate a close relationship between the maximum migration threshold and the signal values of given datasets. We propose in future to ascertain an optimisation function which would learn the maximum migration threshold during run time. This work represents a necessary step towards producing a DCA which automatically interfaces with any given anomaly detection dataset.

Keywords: Artificial Immune Systems · Dendritic Cell Algorithm · Parameter Tuning

1 Introduction

The Dendritic Cell Algorithm (DCA) is an Artificial Immune System (AIS) based on the function and behaviour of the dendritic cells of the human immune system. It is driven by a concept termed the danger theory, which postulates that the human immune system has the ability to discriminate between 'safe' and 'dangerous' contexts[18], informally known as 'danger signals'. The danger theory is in opposition to the classical self-nonself theory of discrimination of antigen proteins via their structure and origin. In the danger theory, and indeed in the DCA, antigen is classified through correlation of context with danger signals and not via examination of the structure of the antigen proteins. The DCA is inspired by this model, as are other similar danger based algorithms[20][16]. Details of the function of the DCA are given in Section 2.

This paper is motivated by a comparison study performed by Lau & Lee in 2018[17], where a direct comparison between the DCA and an artificial neural network (ANN) is performed. This is an innovative application of the DCA, applied as a monitor for human behaviour in carrying out tasks in a VR/AR CUBE setup. The study's results indicate that the DCA can produce a similar good performance on this task. They also indicate that the DCA has a distinct advantage over ANNs when lengthy training periods are required. ANNs have enjoyed a resurgence in popularity in the guise of 'Deep Learning'. This extends on the traditional ANN through adding an optimisation function (frequently gradient descent) to the signal inputs, multiple nodes in multiple layers and a discrimination technique such as softmax to aggregate classification. The implication is that Deep Learning based on ANNs can now tackle 'Big Data' in a computationally feasible manner. Ease of implementation facilitated through TensorFlow and Keras have further increased the popularity beyond the machine learning community. Widespread application of this technique, and the automation of the selection of inputs has heightened the learning capacity of these techniques, now popular in image processing in particular. Given the direct comparison in Lau & Lee[17], we postulate that if the DCA can be directly compared to an ANN, there must be properties of the DCA which make transforming the algorithm into a Deep Learning framework possible at least in principle.

The DCA dispenses with a lengthy training period in favour of expert learning to map input streams. However, multiple authors have attempted to automatically map inputs to the algorithm, as reviewed in Chelly & Elouedi[3].. This already suggests a step towards incorporating a dynamic learning component to this algorithm akin to the first stage in deep learning. However, there are numerous "back-end parameters" in the DCA which have not been subjected to the same automation processes. In this paper we identify a particular parameter which is ideally suited to parameter tuning. Furthermore, Elisa *et al.*[5] apply k-means clustering to the output of the algorithm to refine the discrimination features of the algorithm. A significant improvement is shown in this re-imagined DCA architecture when applied to a standardised intrusion detection dataset, highlighting the importance of the state-change based discrimination performed by the algorithm, indicating the importance of the "back-end parameters".We focus on examining the assignment of the migration threshold across the agents in the DCAs population, to highlight the importance of this parameter's influence on classification accuracy. This is the most basic experiment possible to investigate learning within this algorithm, while maintaining the DCA's key advantage of dispensing with the requirement for a lengthy training period.

The main contribution of this paper is to assess the impact of tuning the maximum migration threshold parameter on the algorithm's classification accuracy. Section 2.1 describes the major features of the algorithm and formal analysis of the algorithm is reviewed in Section 2.2. A rationale for exploring the sensitivity of the maximum migration threshold parameter is given in Section 3. A preliminary experiment including learning the migration threshold parameter is shown in Section 4 comparing a range of values. We conclude the paper by

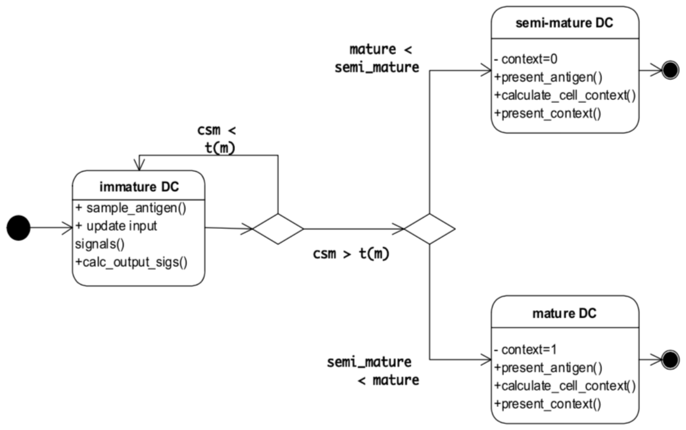


Fig. 1: Schematic representation of the processing by a single population member, demonstrating the migration process, from [6]

suggesting how this modification can be extended to further enhance the DCA towards a Deep Learning DCA framework. For a comprehensive review of the DCA see [3], and the original DCA is described in detail in [6].

2 The Dendritic Cell Algorithm

2.1 Algorithm Overview

As an algorithm, the DCA was first presented in 2005[7] as an anomaly detector in the style of a population based algorithm. Individual DCs in a population are transformed from an immature state to either ‘mature’ or ‘semi-mature’, depending upon the type of ‘signal’ they have encountered throughout their defined lifespan. Expert knowledge couples data streams to the DC population through a rough categorisation of the streams into ‘safe’ or ‘danger’. The stream data is processed by individual DCs though a simple weighted sum equation.The output of this weighted sum increments internal values, either the ‘mature’ or ‘semi-mature’ indicator The data collection window for each DC is determined by a lifespan limit, termed in the literature as ‘migration threshold’[9]. Upon migration a DC is classed as either ‘mature’ or ‘semi-mature’ via the application of a linear threshold or simply labelling the cell based on which of the mature or semi-mature variables is the larger value, demonstrated in Figure 1.Classification cannot be performed with the DCA without an orthogonal data stream termed ‘antigen’ - this is a representation of the item to be classified.

Each member of the set of antigen in a dataset is termed an ‘antigen type’, of which will have multiple instances. This decoupling of the data allows for data correlation within the DCA. The first real-world test for the DCA involved experiments similar to this though monitoring an individual host information and not a network and system calls[8]. Each system call per process is captured in this dataset, and the process ID associated with each system call forms the antigen types in this dataset. This same dataset is used to experiment with the maximum migration threshold in this paper.

In the population each DC agent samples signals from the signal stream and antigen from the antigen stream within a dynamic lifespan. The sampling duration of a DC is controlled by its migration threshold, assigned upon creation of the immature DC. An internal variable of an immature DC, termed in the literature as 'csm'[10], is incremented in proportion to to the strength of signal experienced by the DC. Upon the 'csm' variable having a greater value than the assigned migration threshold, the DC is removed from the sampling pool and presented for analysis. The secondary analysis phase of the algorithm counts for each antigen type the percentage of DCs which are mature versus semi-mature. This ratio returns a value between 0-1 for each antigen type, with values closer to 1 indicating an anomaly, and this is referred to as the 'mean context antigen value' or MCAV. Once all data is sampled a final value for each antigen type is calculated, a linear threshold is applied. A range of values for this threshold can be used to create ROC curves out of the DCA output.

2.2 Theoretical Research and Formal Specification

The prototype version of the DCA was first presented by Greensmith *et al.* in 2005[7], with the full version published in 2006[9]. While implementable details were attempted, the algorithm's function and behaviour were obfuscated by the complex agent based framework used to implement the DCA and the over twenty potentially tuneable parameters. Two approaches were taken to 'demystify' this algorithm and to increase its applicability. The first approach was to reduce the number of tuneable parameters to two in the Deterministic DCA[11] **dDCA**, leaving population size and range of migration threshold across the population. Dynamic antigen buffers, sigmoidal functions for weighted sum inputs, and MCAV was replaced with a real valued metric, K_α.

The dDCA as a simplified algorithm has proven popular for implementation and assisted in some of the earliest theoretical research for the DCA [14]. Aside from the simplification of the algorithm, a key motivator for the development of the dDCA was to provide a 'stripped down' version of the algorithm in order to build in new components, to add in stochastic elements individually. This has not happened to any great extent, though the dDCA has become a studied and applied algorithm e.g.[15] in its own rite as detailed in the review in [3]. Further theoretical analysis of the DCA is performed in Oates *et al.* [19], Stibor *et al.*[21] and Gu *et al.*[13], which analysed the DCA as a set of linear classifiers, without analysing the impact of the antigen stream.

The second attempt to clarify the algorithm is motivated by the inconsistencies in DCA implementations. Ambiguity surrounding signal mapping, the use on inappropriate datasets and direct comparison with unsuitable supervised learning techniques motivated Greensmith & Gale in 2017[12]. A formal specification of the dDCA with Haskell is presented. Haskell is a purely functional language, where the specification becomes the implementation, therefore if the specification is verified as correct, then the implementation is also correct. This research shows that the input to the DCA is stream data, and not necessarily 'feature vectors' and that the 'antigen stream' must be de-coupled from the signal streams in order for the algorithm to be effective. This is the version of the dDCA used in experiments in Section 4 using a verified dDCA. If the learning process for "back-end" tuning is possible, the Haskell specification will be extended to ensure correct future implementation of this component.

3 Cell Migration Control in the DCA

It came as a surprise to reviewers past that there is no explicit learning process, optimisation or local search operator in the DCA. The assumption in the literature is that an AIS must behave like any other evolutionary algorithm. It is thought that it must converge upon a solution like the clonal selection or at least engage in a training process akin to supervised learning techniques in AIS including negative selection[2]. However, the DCA does not have such facility, the deterministic DCA even more so as it dispenses with reliance on any random elements included in the original variant.

The obvious approach is to replace with the requirement to use expert knowledge to decide how signals from the signal stream are mapped to the categories of safe and danger as indeed has been widely performed in the DCA literature, as reviewed in[3], including the use of fuzzy systems, rough set theory and PCA. Secondarily is the optimisations of the weights in the signal processing equation which encompasses a training phase for the DCA, as performed by Elisa *et al.*[4] though the use of a genetic algorithm. This is in contrast to the work presented in [17] which determined that the lack of lengthy training period of the algorithm was indeed how the DCA has an advantage over ANNs.

There are other aspects of the DCA which can be augmented with some form of learning capability outside of the paradigm of requiring a training period. Two tuneable parameters with optimisation potential are **population size** and the assignment of the **maximum migration threshold** of the DC population. The maximum migration threshold is tested for its sensitivity in this paper through performing a preliminary investigation in the link between the parameter and the algorithm's performance. Optimisation of this parameter during the algorithm's runtime may be able to enhance its performance, though this must be done in an incremental manner.

The migration threshold is important in the DCA as it determines the exact set of signal and antigen instances processed within an individual DC throughout the run-time duration of the algorithm. It controls the length of time a DC remains in the sampling pool before being presented for analysis. A migration threshold is assigned to each DC upon the initialisation of the algorithm. In the dDCA, each DC is given a specific value of migration threshold which

Table 1: Example of migration threshold assignment for a population of 5 DCs with a maximum migration threshold Max_{mt} of 10

Cell ID	DC_mt
1	2
2	4
3	6
4	8
5	10

is calculated in proportion to an overall maximum migration threshold, set as a user-definable parameter as $DC_{mt} = f(Max_{mt}/NumCells)$. For example if there are 5 DCs in the sampling pool and a maximum migration threshold of 10 is assigned the DCs migration thresholds are assigned as a simple modulus function as shown in Table 1.

The migration threshold is applied to a parameter termed 'csm' [1] is incremented through summation of the danger and safe signals collected at each signal sampling iteration. Once the value of 'csm' exceeds that of the DC_mt, the cell is removed from the sampling pool and presents data for the analysis phase. At this point, the DC is destroyed, and a new DC is created with an identical DC_mt and repopulated the sampling pool. This process is specified formally in [12].

We commence the investigation into parameter tuning in the dDCA by firstly ascertaining the sensitivity of the algorithm to variation in the migration threshold of the individual cells, controlled by tuning the master maximum migration threshold parameter. The dDCA is useful for this task as it allows for a high degree of reproducibility and traceability of data within the algorithm. An initial specific set of parameter values are chosen with a view to exploring optimisation techniques in future research.

4 Experiments on Migration Thresholds

4.1 Ping Scan Dataset

Ten datasets are created, originally for [9] and used for the sensitivity analysis in [8], based on performing a series of ICMP Ping Scans on a medium scale university network. This data is designed specifically to assess the parameters of the DCA, and not necessarily to capture all of the nuances of network intrusion detection. Given the experiments relate to a DCA parameter, this justifies the use of this dataset in this case. The generated data captured the processes involved

[1] This is based on the biological model of the up-regulation of the CCR7 receptor in natural DCs in response to binding to intracellular molecules of both cell damage and of healthy tissues. This up-regulation attracts the DC away from the potential site of infection and results in its trafficking to the lymph node where is ceases to sample signals and antigen.

Table 2: Summary of **Danger Signal** Data Across 10 Ping Scan Datasets

Dataset	Mean	Stdev	Median	Sum
S1	7.85	15.96	0.40	243.20
S2	4.81	10.45	0.00	317.60
S3	21.28	27.16	2.00	1425.60
S4	27.53	42.57	0.40	1073.60
S5	29.09	41.19	0.00	1134.40
S6	28.55	42.88	0.00	1056.40
S7	30.03	42.72	2.60	1141.20
S8	24.93	37.36	1.20	1072.00
S9	26.36	42.28	0.00	1107.20
S10	26.83	41.95	0.40	1046.40
Mean	**22.73**	**34.45**	**0.70**	**961.76**

Table 3: Summary of **Safe Signal** Data Across 10 Ping Scan Datasets

Dataset	Mean	Stdev	Median	Sum
S1	54.19	41.72	60.00	1680.00
S2	65.10	42.20	86.67	4296.67
S3	50.30	48.48	80.00	3370.00
S4	36.75	45.80	0.00	1433.33
S5	58.46	43.98	70.00	2280.00
S6	51.35	45.53	50.00	1900.00
S7	43.68	47.73	10.00	1660.00
S8	38.14	48.27	0.00	1640.00
S9	57.62	48.43	90.00	2420.00
S10	34.87	41.98	30.00	1360.00
Mean	**49.05**	**45.41**	**47.67**	**2204.00**

during the scan to form the antigen and measured the network attribute of packets per second sent from the machine instigating the scan. The danger signal is the number of packets per second sent, normalised into a range of 0-100. The safe signal is the inverse rate of change of number of packets per second sent also normalised in the range of 0-100. Summary statistics of the signal data for the ten datasets (S1-S10) are shown in Tables 2, 3 and Table 4, including the duration of the monitored session in Table 4.

As part of this data capture exercise, antigen data is also captured. While over 25 processes were active during the scan duration, four 'processes of interest' were identified as making over 100 system calls for the duration of each scan. These are the `bash` process which is the terminal from which the scan is instigated; the `nmap` process used to instigate the scan; the `pts` pseudo-terminal slave process which is a helper process for the nmap process; and `sshd` process which was used to log into the linux terminal from which the data was collected. We expect in the results for the experiments to indicate the `nmap` and `pts` processes

Table 4: Signal Maximums across all datasets, including normalisation of Signal Total Per Dataset Duration

Dataset	Danger (D)	Safe (S)	Sum ($D+S$)	Duration	Normalised SigTotal/ Duration
S1	243.20	1680.00	1923.20	30.81	62.42
S2	317.60	4296.67	4614.27	66.08	69.82
S3	1425.60	3370.00	4795.60	66.96	71.62
S4	1073.60	1433.33	2506.93	38.08	65.84
S5	1134.40	2280.00	3414.40	38.14	89.52
S6	1056.40	1900.00	2956.40	36.62	80.74
S7	1141.20	1660.00	2801.20	37.53	74.64
S8	1072.00	1640.00	2712.00	42.34	64.06
S9	1107.20	2420.00	3527.20	41.14	85.74
S10	1046.40	1360.00	2406.40	38.50	62.50
Mean	**961.76**	**2204.00**	**3165.76**	**43.62**	**72.57**

as anomalous and the `sshd` and `bash` processes to be classified as normal, as indicated in the results in [8]. In previous experiments where the anomaly score is given as the *mean context antigen value* - *MCAV*, a coarse threshold of 0.5 is added to discriminate between the normal and anomalous processes as in [8].

4.2 Experiments

A control experiment is performed with the dDCA using the standard parameters for population size of 100 and the Max_{mt} set at 100. All other settings are as detailed in [11] and [12]. The results for each dataset are shown in Table 5.

Five parameters are chosen on a logarithmic scale to examine the link between the dataset and Max_{mt} set at 1, 10, 100, 1000 and 10000. Given the ranges of the data, this covers the smallest window possible ensuring that the cells will migrate each iteration. The maximum value of 10000 exceeds the total signal amount for each dataset, ensuring that each cell will only migrate once. For the sake of completeness, we also test the average signal sum across all 10 datasets which is 3165. We also test a normalised version of this value which takes into account the duration of each dataset, resulting in a Max_{mt} value of 73. Results of these experiments are shown in Table 6, as mean values per process of interest and the related standard deviation. A more detailed presentation of individual results per process is given in Figure 2. We expect correct classification to produce values of below 0.5 for bash and sshd, and above 0.5 for nmap and pts.

4.3 Discussion

The results clearly show that the maximum migration threshold parameter is important for the dDCA, producing marked changes in classification performance.

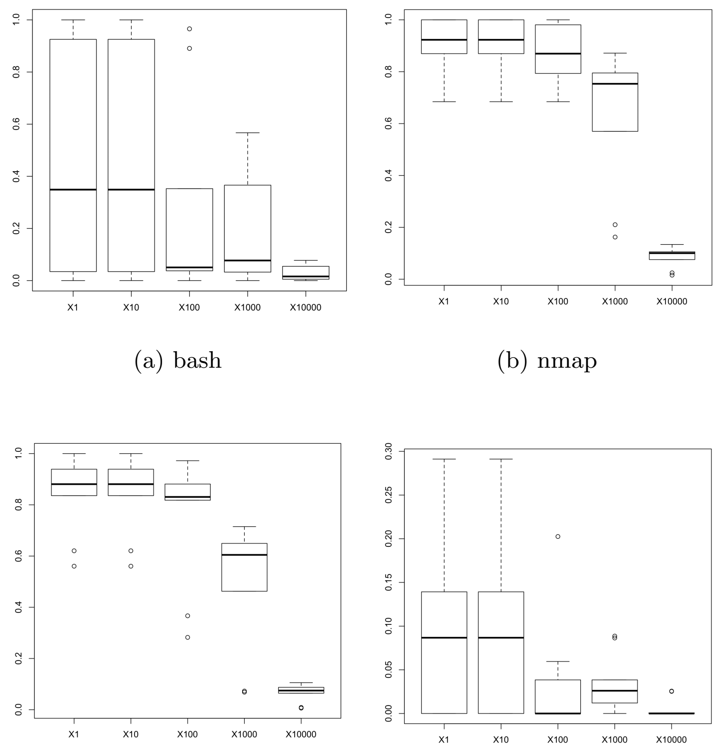
Table 5: Control experiment using previously published parameters: population size = 100; Max_{mt} =100

Dataset	Bash	Nmap	Pts	Sshd
S1	0.00	1.00	0.62	0.08
S2	0.03	0.92	0.56	0.00
S3	0.03	0.93	0.85	0.00
S4	0.37	0.87	0.90	0.09
S5	0.05	0.79	0.86	0.04
S6	0.76	0.68	0.87	0.05
S7	0.53	0.97	0.93	0.14
S8	1.00	1.00	1.00	0.29
S9	0.05	0.87	0.86	0.04
S10	0.93	1.00	0.88	0.00
Mean	**0.38**	**0.90**	**0.83**	**0.07**

Table 6: Results of parameter variation of Maximum Migration Threshold, including mean across all 10 datasets, and accompanying standard deviation. The value of 3165 represents the mean signal per session, and 73 is the mean signal per session normalised by the number of signal instances.

	Bash	Bash	Nmap	Nmap	Pts	Pts	Sshd	Sshd
Max Migration	mean	stdev	mean	stdev	mean	stdev	mean	stdev
1	0.44	0.44	0.90	0.10	0.84	0.14	0.08	0.09
10	0.44	0.44	0.90	0.10	0.84	0.14	0.08	0.09
100	0.25	0.37	0.87	0.11	0.75	0.23	0.03	0.06
1000	0.17	0.20	0.64	0.25	0.50	0.23	0.03	0.03
10000	0.03	0.03	0.07	0.04	0.06	0.03	0.00	0.010
3165	0.08	0.08	0.27	0.12	0.21	0.10	0.02	0.03
73	0.44	0.43	0.90	0.10	0.84	0.14	0.08	0.09

The result that identical $MCAV$s are obtained for all values under 100 was initially surprising, and assumed to be a fault in the experimental test harness. Thorough investigation of this phenomena was performed as a result, and we are confident that this is a genuine observation and not due to a bug in the dDCA code or in the test harness. Upon analysis, we see that for each signal instance a combined value of 100 is present. This means that for instance, for cells with a migration threshold of less than 100, all cells in the population migrate, making parameter variability in this range immaterial. This is a useful observation for future guidance on setting this parameter. As the parameter increased above 1000, there is a deterioration in the discrimination of the anomalous processes, though changes in the discrimination of the normal processes were not significant. This is most pronounced with the value of 10000 in which no migration occurs until all signal instances are processed, as shown in Figure 2. These results indicate that this parameter and the migration thresholds of individual DCs



(a) bash (b) nmap

(c) pts (d) sshd

Fig. 2: Boxplots of the four processes of interest, showing the MCAV for each of the parameter settings per process

within the population does warrant further investigation and the application of an optimisation method.

4.4 Conclusions

The contribution of this paper is that it shows the sensitivity of the migration threshold parameter in the dDCA and on a wider range of data than in previous experiments with the dDCA. Deterioration of classification is shown with excessively large migration thresholds, and lower limits related to the current system signal values. Therefore this represents a small but important step towards implementing a learning mechanism in the DCA independent of an initial training phase. The results suggest that the maximum migration threshold is likely to benefit from an optimisation technique either based on the expected input for the algorithm or, more importantly, during run time. This can be achieved via lightweight local search operator and we hope to explore this in subsequent studies. We have not studied the distribution of the migration thresholds across the population as a uniform distribution is used here. Pertinently, a uniform distribution is used in this paper, and we do not know how the results would be affected if for example a gaussian distribution be used in its place.

In this paper we have ignored the potential influence of the number of cells in the population. Therefore a multi-objective optimisation approach to tune both key parameters of the dDCA may be beneficial in ascertaining their optimum values for any given dataset. We are aware that there may also be nuances of this particular dataset which are influencing the results, and therefore we would seek to replicate this study on a different dataset, for example using the KDD99 dataset as a starting point. The central goal is to move the DCA towards a Deep Learning style framework, and understanding the influence of the migration threshold is just one component which contributes to this aim. An integrated approach examining both front and back-end parameters in a dependent fashion would be the intended trajectory of future work on this algorithm. There is also the potential to run multiple DCA instances in parallel in a similar multilayered fashion to an ANN. A combination of these techniques will be needed to achieve the aim of creating a Deep Learning DCA.

References

1. de Castro, L., Timmis, J.: Artificial Immune Systems: A New Computational Approach. Springer-Verlag, London. UK. (September 2002)
2. Chelly, Z., Elouedi, Z.: A survey of the dendritic cell algorithm. Knowledge and Information Systems **48**(3), 505–535 (September 2016). https://doi.org/10.1007/s10115-015-0891-y
3. Elisa, N., Yang, L., Naik, N.: Dendritic cell algorithm with optimised parameters using genetic algorithm. In: 2018 IEEE Congress on Evolutionary Computation, CEC 2018, Rio de Janeiro, Brazil, July 8-13, 2018. pp. 1–8 (2018). https://doi.org/10.1109/CEC.2018.8477932, https://doi.org/10.1109/CEC.2018.8477932
4. Elisa, N., Yang, L., Qu, Y., Chao, F.: A revised dendritic cell algorithm using k-means clustering. In: 20th IEEE International Conference on High Performance Computing and Communications; 16th IEEE International Conference on Smart City; 4th IEEE International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2018, Exeter, United Kingdom, June 28-30, 2018. pp. 1547–1554 (2018)
5. Greensmith, J.: The Dendritic Cell Algorithm. Ph.D. thesis, School of Computer Science, University Of Nottingham (2007)
6. Greensmith, J., Aickelin, U., Cayzer, S.: Introducing Dendritic Cells as a novel immune-inspired algorithm for anomaly detection. In: Proc. of the 4th International Conference on Artificial Immune Systems (ICARIS), LNCS 3627. pp. 153–167. Springer-Verlag (2005)
7. Greensmith, J., Aickelin, U., Tedesco, G.: Information fusion for anomaly detection with the DCA. Information Fusion **11**(1), 21–34 (2010)

8. Greensmith, J., Aickelin, U., Twycross, J.: Articulation and clarification of the Dendritic Cell Algorithm. In: Proc. of the 5th International Conference on Artificial Immune Systems (ICARIS), LNCS 4163. pp. 404–417 (2006)
9. Greensmith, J., Twycross, J., Aickelin, U.: Dendritic cells for anomaly detection. In: Proc. of the Congress on Evolutionary Computation (CEC). pp. 664–671 (2006)
10. Greensmith, J., Aickelin, U.: The deterministic dendritic cell algorithm. In: Artificial Immune Systems, pp. 291–302. Springer (2008)
11. Greensmith, J., Gale, M.B., IEEE: The functional dendritic cell algorithm: A formal specification with Haskell. In: 2017 IEEE Congress on Evolutionary Computation, CEC 2017, Donostia, San Sebastián, Spain, June 5-8, 2017. pp. 1787–1794. IEEE (2017)
12. Gu, F., Feyereisl, J., Oates, R., Reps, J., Greensmith, J., Aickelin, U.: Quiet in class: classification, noise and the dendritic cell algorithm. In: Artificial Immune Systems, pp. 173–186. Springer (2011)
13. Gu, F., Greensmith, J., Aickelin, U.: Theoretical formulation and analysis of the deterministic dendritic cell algorithm. Biosystems **111**(2), 127–135 (2013)
14. Igbe, O., Darwish, I., Saadawi, T.: Deterministic dendritic cell algorithm application to smart grid cyber-attack detection. In: 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud). No. New York, NY, USA, IEEE (2017)
15. Kim, J., Bentley, P., Wallenta, C., Ahmed, M., Hailes, S.: Danger is ubiquitous: Detecting malicious activities in sensor networks using the dendritic cell algorithm. In: Proc. of the 5th International Conference on Artificial Immune Systems (ICARIS), LNCS 4163. pp. 390–403 (2006)
16. Lau, H.Y.K., Lee, N.M.Y.: Danger theory or trained neural network - A comparative study for behavioural detection. In: Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS), Toyama, Japan, December 5-8. vol. 10.1109/SCIS-ISIS.2018.00143, pp. 867–874 (2018)
17. Matzinger, P.: Tolerance, danger and the extended family. Annual Reviews in Immunology **12**, 991–1045 (1994)
18. Oates, R., Kendall, G., and, J.G.: Frequency analysis for dendritic cell population tuning: Decimating the dendritic cell. Evolutionary Intelligence: Special Issue on Artificial Immune Systems (2008)
19. Sarafijanovic, S., Boudec, J.L.: An artificial immune system for misbehavior detection in mobile ad-hoc networks with virtual thymus, clustering, danger signal and memory detectors. In: Proc. of the 3rd International Conference on Artificial Immune Systems (ICARIS), LNCS 3239. pp. 342–356 (2004)
20. Stibor, T., Oates, R., Kendall, G., Garibaldi, J.M.: Geometrical insights into the dendritic cell algorithm. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation. pp. 1275–1282. ACM (2009)