Rule Networks in Learning Classifier Systems

Karthik Kuber, Stuart W. Card, Kishan G. Mehrotra, and Chilukuri K. Mohan
Dept. of Electrical Engineering and Computer Science, Syracuse University
Syracuse, NY, USA
{kkuber,swcard,mehrotra,mohan}@syr.edu

ABSTRACT

Interrelationships between rules can be used to develop network models that can usefully represent the dynamics of Learning Classifier Systems. We examine two different kinds of rule networks and study their significance by testing them on the 20-mux problem. Through this experimentation, we establish that there is latent information in the evolving rule networks alongside the usual information that we gain from the XCS. We analyze these interrelationships using metrics from Network Science. We also show that these network measures behave as reliable indicators of rule set convergence.

Categories and Subject Descriptors

G.2.2 [**Graph Theory**]: Network problems; I.2.6 [**Learning**]: Knowledge Acquisition; I.2.11 [**Distributed Artificial Intelligence**]: Intelligent agents

General Terms

Network Science Applications, Evolutionary Algorithms, Learning Classifier Systems

Keywords

Evolutionary Algorithms, Genetic Algorithms, Learning Classifier Systems, XCS, Network Science, Convergence Detection

1. INTRODUCTION

In studies of evolutionary algorithms, relationships between individuals are usually ignored. Some efforts have focused on relationships between subsets of the population. Our focus is on examining relationships between individuals, asking what happens over time to the structural properties of networks representing evolving populations. Networks are important to study because of the following reasons: (i) Networks take into account relationships between individuals (via the existence of edges and their lengths), (ii) Networks are better suited to depict various other kinds of relationships as well, such as ancestral connections, and (iii) Networks provide useful models of relationships that biological species share.

In most other network-theoretic applications where networks change over time, the nodes in a network are mostly the same from one instant to the next, although some edges (and a small number of nodes) may appear or disappear. However, in applying this approach to evolutionary algorithms, we observe that individuals in an evolving population may not persist over time. Nodes in a network vanish in one generation, so that there is no obvious "continuity" between networks representing successive generations. Hence the key questions to be posed are at the level of the entire network (representing the entire population) or its subgraphs (representing sub-networks consisting of "related" individuals).

We present below a brief introduction to networks.

1.1 Introduction to Networks

The study of graphs and networks has been ongoing for a long time. However, the study of dynamic networks has been exciting considerable interest in recent years, particularly in the social networks context, mostly because of the success and popularity of many online social networks such as Facebook, Twitter, and LinkedIn. A comprehensive introduction to Network Science is provided by Newman [10].

Although the terms "graphs" and "networks" may be used interchangeably in many cases, a graph is an abstract representation of a network. For instance, a user's friends and social connectivities constitute a *network*, but their representation is a *graph*. Similarly, in our case, the population that is performing an optimization is a network of individuals, but is represented as a graph.

Any graph consists of a set of vertices (or nodes), $\mathbb{V}$, and a set of edges $\mathbb{E}$. An edge between two nodes represents some form of connection between them. The edge can be directed, implying that there is an asymmetric relation between the nodes, or undirected, implying a symmetric one. One example of a directed graph models the Twitter network where if a person A follows a person B, it does not imply that B follows A; for example, a celebrity on Twitter may have many more followers than he/she is following. In the case of an undirected graph, such as in Facebook, if a person A is a friend of person B, it implies that B is also a friend of A. This establishes an undirected edge between the nodes representing the two people.

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.

ACM 978-1-4503-2881-4/14/07 ...$15.00.
http://dx.doi.org/10.1145/2598394.2611382.

1.2 Selected Network Terminology

Some frequently used terms associated with networks are:

- Degree: The degree of a node is the number of edges connected to it. Directed networks have an in-degree and an out-degree, although in this paper we consider only undirected networks.
- Weighted Networks: Unlike unweighted networks, where all edges are considered equal for analysis, in certain types of networks, edges can be assigned a weight, if there is a need to represent more than just the presence or absence of edges. This is typically a real number, and it can represent various measures in different kinds of networks. For example, when modeling networks of friendships, we might assign stronger weights to edges between close friends, and weaker ones with acquaintances. We can assign each edge a certain weight indicative of this difference in friendship levels.
- Community: A community is a subgraph which is characterized by a tightly-knit structure. Each node in a community is connected to many others within that community, and significantly fewer edges between communities. A network is said to have a community structure if it can be sub-divided into such sub-networks (with or without overlaps).
- Adjacency Matrix: This is a two-dimensional matrix with vertices representing the rows and columns and the $(i, j)^{th}$ entry representing the presence or absence of an edge between vertices i and j. This matrix consists of 1s and 0s in case of an unweighted network, i.e., the presence or absence of an edge, and real values in the case of a weighted one, i.e., the strength of the edge. Also, the matrix is symmetric across the principal diagonal if the network is undirected, and asymmetric if directed.
- Component: If there exists a path from every node to every other node in a network, then the network is said to be fully connected. A sub-graph displaying similar characteristics is a component, i.e., there is a path from each node to every other node, but not to other nodes in the network that are not part of this component. Here, we are interested in a component of rules (nodes).
- Large Component: If the size of a network component grows in proportion to the number of nodes in the network, it is called a "Giant Component" [10]. When the network is not associated with a specific growth process, we refer to a component that contains a large proportion (say $80-90\%$) of the nodes in the network as a "large" component.

We explain some of the above terms with an example network in Figure 1. This is a *single* unweighted, undirected network consisting of 9 nodes. This could, for example, be a network of 9 rules in an LCS, with an edge denoting some kind of interrelationship between them.

There are two components, one composed of nodes $5, 6, 7, 8$ and the other having nodes $1, 2, 3, 4, 9$.

The degrees of nodes $1-9$ are: $\{3, 2, 3, 1, 1, 3, 1, 1, 1\}$ respectively. When normalized to a range $[0, 1]$, the degrees

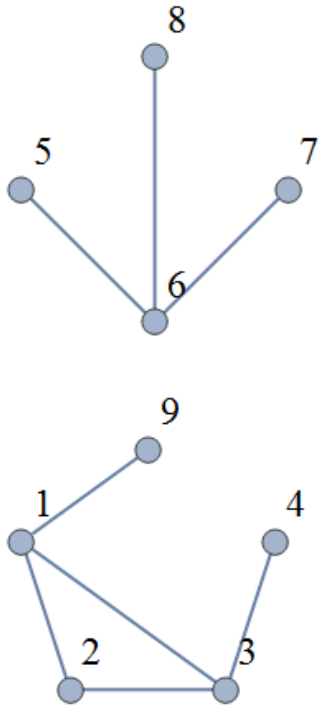

Figure 1: Network Example

are $\{1, 0.67, 0.33, 0.33, 0.33, 1, 0.33, 0.33, 0.33\}$, i.e., the highest degree gets the value 1, and the others degrees are computed proportionally.

The adjacency matrix would be a 9×9 matrix of 0s and 1s, with a 1 in a cell indicating an edge between the nodes corresponding to the row and column of that cell.

Many other metrics, measures and examples are discussed in [10].

2. RELATED WORK

2.1 Networks in the context of EAs

Genetic Algorithms and other population-based evolutionary algorithms are rich sources of information contained in the ever-changing generations and populations. The effects of genetic operators and other ideas borrowed from observing various biological species serve as models to study the growth of a species from its infancy to maturity as a community or communities. Observing that the natural behavior of most species in the biosphere is to form cohesive communities (or networks) for survival and growth, we believe that a deeper understanding of such evolutionary systems can be greatly enhanced by the science of networks.

Considering that each generation of an evolving population contributes towards one snapshot of a network, we have, in effect, created a dynamic network with a simple biological evolution system.

In prior work, we had begun the exploration with a study of networks in Genetic Algorithms [6]. In that work, we explored the notion of individuals in an EA having underlying interrelationships which we experimentally found to be of use in understanding and improving the performance of an EA. The main focus was on networks created using a con-

straint on the Euclidean distance between nodes. In other words, a Euclidean Network $G =< \mathbb{V}, \mathbb{E} >$ was defined as follows:

- $\mathbb{V}$: All n individuals in the population are vertices or nodes.
- $\mathbb{E}$: Two individuals have an edge between them if the Euclidean distance between them is less than the "Edge Threshold" value (δ). This value was chosen to be proportional to the length of the body-diagonal of the search space, since the "closeness" of two points is relative to the space being searched.

In that work, we had found that:

1. Analyzing a GA via network metrics can help in identifying convergence faster.
2. We can use these network measures and metrics to speed up a GA.
3. We can use these evolving networks to identify a niche space where the optimum solution lies, based on experiments performed on five test problems .
4. The largest component of the network is a key identifier of the region of interest and its centroid is a good representative of this space.

Summarizing the above, we found that there is benefit in analyzing GAs as evolving networks instead of evolving populations. In the work so far, the populations we have been considering have been real-valued. We wish to explore the behavior of individuals and networks in a specialized evolutionary system. We choose to study networks in Learning Classifier Systems [7], more specifically with XCS [1][2][3][14]. We wish to extend the concept of analyzing EAs using networks by creating rule networks in an LCS, and studying the evolution of network parameters as well as XCS evaluation measures and examine their similarities.

Some earlier work involving the study of rules in rule-set form, includes the creation of rule clusters in an XCS system to specialize rule-sets and to merge rules which overlap, and the removal of redundant rules [11]. Attribute co-occurrence networks in LCSs have been studied in [13]. Also, an information theoretic approach to rule specialization involves the study of the development of specialized rule-sets from a mutual information viewpoint [12].

We proceed to define the similarity between rules and the creation of a network of rules.

3. SIMILARITY MEASURE

Since the genotypes of the rules have bit-string representations, a similarity measure based on the Hamming distance can be applied, as in [8]. We define the similarity between two rules similar to the Hamming distance where we make bitwise comparisons. But these rules contain don't cares (#), therefore we not only consider the number of differences, but also by how much. In other words, we define similarity between pairs of individuals in a ternary system involving a don't care as:

- Similarity between 1 and 1 or 0 and 0 is 1.
- Similarity between 1 and # as well as that between 0 and # is 0.5. This is because in each of these cases, there can be two values for each don't care (0 or 1) where one of the values is a perfect match having a similarity of 1, and the other is a mismatch, an exact opposite with a similarity of 0. Hence, we use the average value which is 0.5.
- Similarity between # and # is 0.5. This is because there are four combinations here: If both are the same (both 0 or 1), then there is a perfect match with a similarity of 1. Else, they are negations of each other, implying complete dissimilarity.

The more similar the rules are, the stronger the edge. In this computation, we compare only the condition part of the rules and not the action, since each incoming point is matched only with the condition string to trigger a rule.

4. NETWORK CONSTRUCTION

We create networks in two different ways:

1. A weighted network: An edge is assigned a weight proportional to the similarity between the two vertices. Hence, the edge strengths can vary from 0 (i.e., no edge), to C_L, i.e., an exact match. (Here, C_L is the length of the multiplexer problem being solved and is equal to $k + 2^k$ for the multiplexer problem, where $k \in I^+$. k represents the number of select lines of the multiplexer, and 2^k refers to the possible binary strings input to the multiplexer.)
2. An unweighted network: This network has a similarity threshold for two rules. In other words, two nodes are connected by an edge if the edge strength is greater than a threshold. Hence, all nodes are not connected to each other, unlike the previous case. In this context, we perform a degree-based analysis.

We examine rule-set convergence indicators from both of these methods.

5. EXPERIMENTS AND RESULTS

In order to correlate the network measures, we use the following two popularly used XCS evaluation measures [14]:

- Accuracy (or Performance): The fraction of the last 50 trials that were correctly predicted.
- System Error: Normalized absolute difference between system prediction and external payoff, averaged over the last 50 trials.

Hence, the goal is to improve accuracy while reducing system error. Performance analysis of an LCS is also discussed in detail in [4] and [5].

We will compare these XCS evaluation parameters with the network parameters that we will describe next. These network parameters are used to detect rule-set convergence from a network perspective. An alternate view of rule-set convergence based on parent-trees of rules has been studied in [9].

5.1 The Weighted Network

In the first network, we have a completely connected network, but with varying edge weights. Network measures as well as XCS measures are plotted in a dual Y-axis against Generation on the X-axis in Figure 2, we observe that there

is a rise and fall of the average edge weight per rule, akin to the component structures seen in a GA [6], and their collapse. If a function is monotonically increasing or decreasing, without knowledge of the distribution, it is hard to determine when it will stabilize. But by having a rise and fall structure, we have a baseline to work with.

Further, we note the strong correlation between the average edge weight per rule (and also the same measure per microclassifier), and the XCS evaluation measures of Accuracy and System Error. These graphs are plotted for every 50^{th} generation, and the values are averages over the past 50 generations.

In Table 1, we list the observations at various generations. We also compute two measures of correlations over all data from the experiments:

1. Correlation between the System Error and the Average edge weight per microclassifier is 0.848, and
2. Correlation between the System Error and the Average edge weight per rule is 0.857

Both correlations have p-values < 0.001, which are indicative of strong correlations between the XCS evaluation measures and the corresponding network measures.

5.2 The Unweighted Network

In the unweighted network, we set a threshold of half the maximum similarity possible as the criterion for the existence of an edge. Creating the network thus, this time we plot the distributions of degrees (after normalization to the range $[0, 1]$), i.e., the degrees of all rules in the network, normalized to account for different numbers of rules in different generations. We align boxplots of these distributions and the XCS measures, and plot them against the Generation, until $50,000$, on the X-axis as seen in Figure 3. Here, too, except the degree distributions (which are plotted at those instants), all values are averaged over the past 50 runs.

From Table 2 and Figure 3, we observe that once the boxplots stabilize, there is negligible change in the System Error. We let the experiment run for about $100,000$ generations and note that in the second half, there is very little improvement as is noticed from the various parameters. In other words, if we had observed the run purely from the network theoretic standpoint, we could have stopped the run reasonably with little accuracy sacrificed by Generation $30,000$, which results in System Error a little higher than in Generation $100,000$ (which was about 5.9).

We also observe from Figure 3 that the degree distribution is very well aligned with the accuracy/system error plots. This is yet another indication of network measures being very well synchronized with EA measures.

6. SUMMARY

In a bit-string representation as seen in an LCS, the similarities between rules are high (which is dependent on the problem being solved and the input points at each generation) when we compare bit-positions. The weight threshold (to determine when an edge exists in the network) can be modified for the problem at hand. More importantly, we notice a strong correlation between the edge weights and node degree distributions and the corresponding convergence criteria in the XCS algorithm (high accuracy as well as low system error).

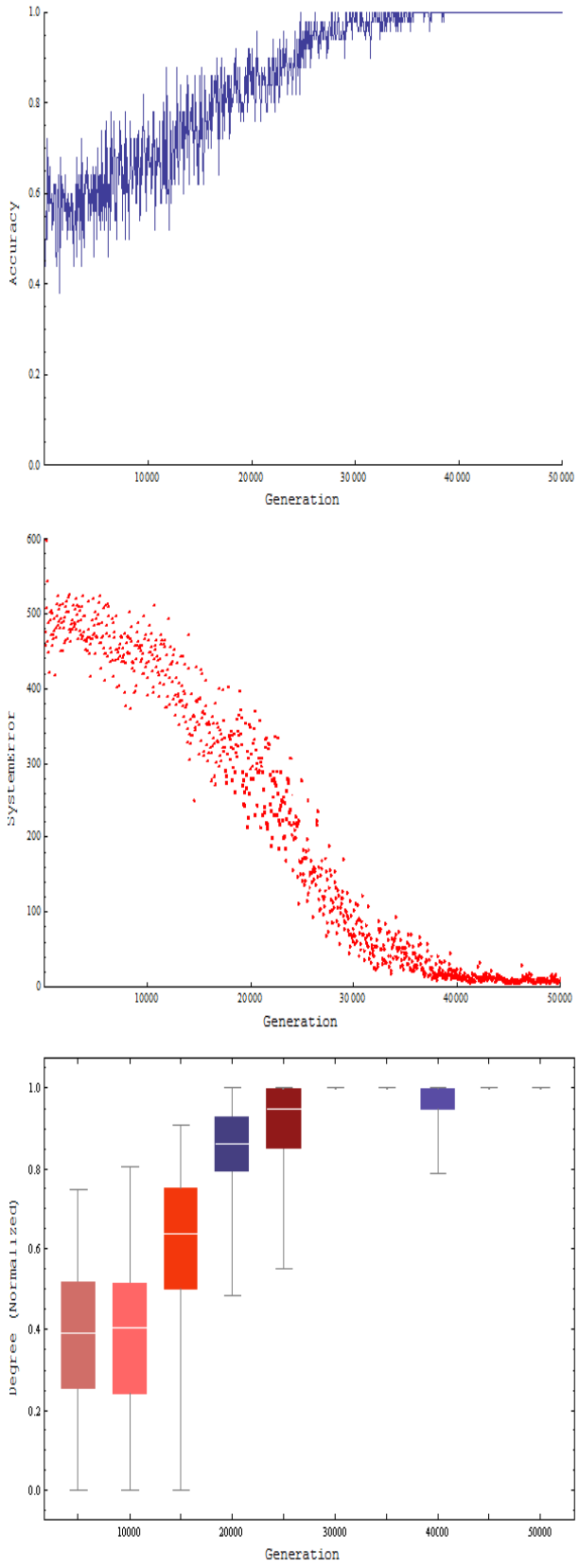

Figure 3: Accuracy, System Error compared with Degree Distributions in XCS: 20-mux

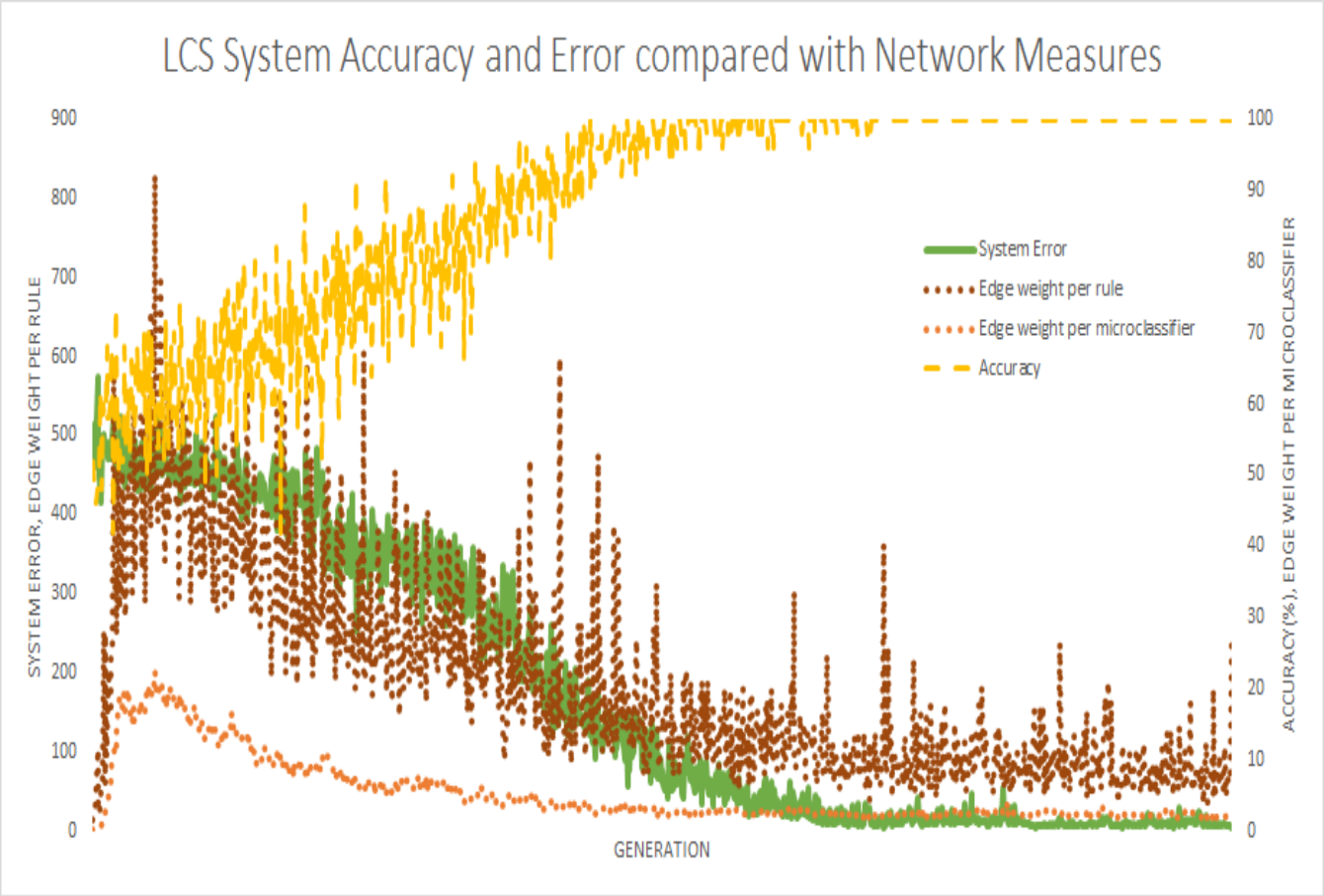

Figure 2: Accuracy, System Error compared with the Edge weights in XCS: 20-mux

Table 1: XCS Evaluation measures: Accuracy, System Error compared with Edge weight per microclassifier and Edge weight per rule

Generation	Accuracy	System Error	Edge weight per microclassifier	Edge weight per rule
5000	0.6	467.658	13.564	541.320
10000	0.6	425.089	8.570	260.379
15000	0.78	303.681	5.934	244.428
20000	0.96	174.084	3.696	164.941
25000	1	61.981	2.422	118.396
30000	1	28.587	2.648	144.453
35000	1	12.024	2.345	98.749
40000	1	55.630	2.884	84.737
45000	1	6.185	2.128	72.584
50000	1	5.182	2.323	241.563
60000	1	10.316	2.311	52.768
70000	1	18.228	1.858	55.337
80000	1	5.059	1.974	54.883
90000	1	6.127	2.435	222.440

Table 2: XCS Evaluation measures: Accuracy, System Error compared with Average Rule Degree and its Standard Deviation

Generation	Accuracy	System Error	Avg. Rule Degree	Rule Std. Dev.
5000	0.66	445.507	0.399	0.166
10000	0.7	414.770	0.380	0.183
15000	0.86	312.837	0.582	0.211
20000	0.92	292.491	0.825	0.139
25000	0.94	168.776	0.905	0.119
30000	0.98	59.863	1	0
35000	1	45.090	1	0
40000	1	23.420	0.974	0.051
45000	1	6.714	1	0
50000	1	9.198	1	0
60000	1	7.546	1	0
70000	1	20.975	1	0
80000	1	13.618	1	0
90000	1	8.356	1	0

Although it is hard to determine when the system error will reach 0, if at all, it is easier to observe the network measures which clearly converge, which is noticed far more easily in the case of degree distributions. For the same parameters, we continued running the XCS for another $50,000$ generations, but the graph was a continuation of the values seen from about $30,000 - 50,000$ and are hence not plotted.

Through these experiments, we observe that although the LCS is a completely different EA system (that involves ternary strings, has many other processes alongside a GA, and involves multi-layered individuals – macroclassifiers and microclassifiers), we are still able to identify and draw parallels with the interrelationships that they share in space. Strong correlations between the network measures and the LCS evaluation measures suggest that the networks and the EA are in synchrony, i.e., the dynamical changes in network and EA measures are very well synchronized.

7. CONCLUSION

In this work, we explore networks that model rule sets and the information they provide about an XCS run. It is of interest to note the amount of information that is conveyed by the underlying rule networks. The strong correlation between network and LCS evaluation measures also suggests that there is underlying interrelationships between the rules that can be tapped.

We plan to study specialization of rule-sets using this additional network information, which can potentially lead to more expressive rule-sets akin to those found information-theoretically in [12]. We propose to retain representatives of rule-sets upon detection of convergence and use the remaining rules to explore other rule spaces, in order to attempt to reduce system error. This is presently being researched and is a subject for future work.

8. REFERENCES

[1] M. V. Butz. 2000. XCSJava 1.0: An Implementation of the XCS classifier system in Java. Technical Report 2000027, Illinois Genetic Algorithms Laboratory

[2] M. V. Butz. 2000. XCS implementation in Java, version 1.0 (Source Code: http://www.illigal.uiuc.edu/pub/src/XCSJava/XCSJava1.0.tar.Z). Retrieved Apr 4, 2009.

[3] M. V. Butz and S. W. Wilson. 2002. An Algorithmic Description of XCS. Journal of Soft Computing, 6 (2002) 144–153.

[4] M. V. Butz. 2005. Rule-based Evolutionary Online Learning Systems: A Principled Approach to LCS Analysis and Design. Vol. 191. Springer.

[5] J. Drugowitsch. 2008. Design and Analysis of Learning Classifier Systems: A Probabilistic Approach. Vol. 139. Springer.

[6] K. Kuber, S. W. Card, K. G. Mehrotra, and C. K. Mohan. 2012. A Network Theoretic Analysis of Evolutionary Algorithms. In Swarm, Evolutionary, and Memetic Computing. Lecture Notes in Computer Science 7677, pp. 585-593. Springer Berlin Heidelberg.

[7] J. H. Holland and J. S. Reitman. 1978. Cognitive Systems Based on Adaptive Algorithms. In D. Waterman & F. Hayes-Roth (eds) Pattern-directed Inference Systems. Academic Press

[8] J. Horn, D. E. Goldberg, and K. Deb. 1994. Implicit Niching in a Learning Classifier System: Nature's Way. Evolutionary Computation 2 (1): 37-66.

[9] M. Iqbal. 2014. Improving the Scalability of XCS-based Learning Classifier Systems. Ph.D. Thesis, Victoria University of Wellington.

[10] M. E. J. Newman. 2010. Networks: An Introduction. Oxford University Press.

[11] L. D. Shi, Y. H. Shi, Y. Gao, L. Shang, and Y. B. Yang. 2011. XCSc: A Novel Approach to Clustering with Extended Classifier system. International Journal of Neural Systems 21 (01): 79-93.

[12] R. E. Smith and M . K. Jiang. 2007. MILCS: A Mutual Information Learning Classifier System. In Proceedings of the 2007 GECCO Conference Companion on Genetic and Evolutionary Computation. London, United Kingdom: ACM, pp. 2945-2952.

[13] R. J. Urbanowicz, A. Granizo-Mackenzie, and J. H. Moore. 2012. An Analysis Pipeline with Statistical and Visualization-guided Knowledge Discovery for Michigan-style Learning Classifier Systems. Computational Intelligence Magazine, IEEE 7 (4): 35-45.

[14] S. W. Wilson. 1995. Classifier Fitness based on Accuracy. Evolutionary Computation 3 (2): 149-75.