Enable the XCS to Dynamically Learn Multiple Problems: A Sensor Tagging Approach

Yao-Ming Wu
Institute of Biomedical Engineering, National Chiao Tung University, Taiwan, R.O.C..
mark183396@hotmail.com.tw

Liang-Yu Chen
Institute of Biomedical Engineering, National Chiao Tung University, Taiwan, R.O.C..
b60306@hotmail.com

Po-Ming Lee
Institute of Computer Science and Engineering, Department of Computer Science, National Chiao Tung University, Taiwan, R.O.C..
kamisan118@gmail.com

Tzu-Chien Hsiao
Biomedical Electronics Translational Research Center and Biomimetic Systems Research Center, Institute of Biomedical Engineering, Department of Computer Science, National Chiao Tung University, Taiwan, R.O.C..
labview@cs.nctu.edu.tw

ABSTRACT
The field of presentation of Extended Classifier System(XCS) has undergone many fluctuations and shifts over the years to adapt different domain problems. With the increasing usage of application of artificial intelligence requirements for more complexity presentations of XCS have become more critical. And learning multiple problems is a subject that was needed but few one studied. To dynamically learn multiple problems during a single learning process, we applied a novel representation of classifier conditions of the XCS, named Sensory Tag (ST) to achieve this goal. The XCS with the ST as the input representation is called XCSSTC. The experiments of the proposed method were conducted for the single step problems. The potential of XCSSTC's use in future application of artificial intelligence clearly needs further exploration.

Categories and Subject Descriptors
F.1.1 [**Models of Computation**]: Genetics-Based Machine Learning, Learning Classifier Systems

General Terms
Algorithms, Performance.

Keywords
Learning Classifier Systems; XCS, Scalability; Pattern Recognition

1. INTRODUCTION
Traditional CSs use binary strings of fixed-length as its representation of classifier conditions. Only a few isolated efforts have continued to address the representation of XCS that can adapt to more domains. But they often focus on learning one kind of problem at the same time. For instance, Wilson [1] modified the classifier condition of the XCS by adopting real-value representation. The revised version is called XCSR [1]. This modification enables the XCS to be applied to continuous variables such as stock index, temperature, height and weight. Furthermore, Lanzi firstly implemented XCS with two different kinds of representations for population-based conditions: One is a variable length messy coding [2], and another is S-expressions [3].

GECCO'15 Companion, July 11-15, 2015, Madrid, Spain
ACM 978-1-4503-3488-4/15/07.
http://dx.doi.org/10.1145/2739482.2764685

The purpose of this study was to improve the capability of XCS that can learn multiple problems at a same time to cope with problems that are more complex in real world in the future. To dynamically learn multiple problems during a single learning process, we applied a novel representation of classifier conditions of the XCS, named Sensory Tag (ST) to achieve the goal. The concept of the proposed method is inspired from the messy coding proposed by Lanzi in 1999 [2]. We proposed the XCS with this implementation of ST as the input representation called XCSSTC [7], [8], [9]. To examine the performance of the XCSSTC in dynamically learning multiple problems during a single learning process, single-step learning problems, that were, the Multiplexer (MUX), the Majority-On (MO), and the Even-Parity problem domain (EP), were used to conduct experiments.

2. DESCRIPTION OF THE XCSSTC
The method we proposed here is to use the Hash Table (HT) [4] to implement the concept of STs as the representation of classifier conditions. The HT stores a collection of (ST, SV) pairs in which each pair has a key and a value. The keys in an HT are unique, and the HT is capable of retreving a value by using its corresponding key efficiently from the itself based on an implementation of a hash function. In the proposed XCSSTC, the ST of a sensor is taken as a key whereas its Sensory Value (SV) is taken as the value corresponds to the key. Each classifier has its own HT as the representation of classifier conditions. The HT includes all the (ST, SV) data pairs. The condition bit which is # in a classifier of an ordinary XCS is simply ignored from the HT for the classifiers in the XCSSTC. For instance, in a 6-bit MUX problem, assumes that the condition of a classifier in an ordinary XCS is 01#1##, the classifier in XCSSTC stores only the set of data pairs $\{(D_0, 0), (D_1, 1), (D_3, 3)\}$ in its HT. The detail mechanisms of the XCSSTC that are different from the original XCS are described as follows.

3. Experimental Description
This section explains the experimental design and the problem domains used in the current study to prove the capability of the XCSSTC in learning multiple problems in a single learning process.

3.1 Research Problems
To prove the capability of the XCSSTC in learning multiple problems in a single learning process, we'd like to experiment and seeks to answer the research questions #1 ~ #8 listed below:

#1 ~ #4 We tested the XCSSTC by having it learn multiple problems at the same time in a consecutive manner. That is, for instance, the first 6-bit MUX (i.e. $MUX6_1$), then changes to the second 6-bit MUX (i.e. $MUX6_2$), then the third one, …, until reached the last 6-bit MUX (i.e. $MUX6_n$), then return to the $MUX6_1$ and all over again. When a problem is chosen, for instance, $MUX6_3$, the sensory inputs of the other problems are considered missing and do not send to the XCSSTC, leaving only the sensory inputs that corresponds to the $MUX6_3$ valid.

#5 ~ #8 How about the same research question #1~#4 but with the problem presentation sequenced controlled? The current study tested this issue by continuously presenting a problem (e.g., $MUX6_1$) until (e.g., for 5k learning instances) the XCSSTC reaches its optimal performance then changed to the next one (e.g., $MUX6_2$), …, until reached the last problem (i.e. $MUX6_n$) and all over again to see if the XCSSTC remains the optimal performance. Table 1 shows an illustration on what we've tested in our experiment.

Table 1 An illustration of the multiple problems we used in our experiment

Problems\Trials	Trial1	Trial2	Trialn
(1)(5)MUX6s	{MUX6,MUX6}	{MUX6,MUX6,MUX6}	{MUX6,MUX6…}
(2)(6)MUX6+MUX11	{MUX6,MUX11}	{MUX6,MUX6,MUX11,MUX11}	{MUX6,MUX11…}
(3)(7)MUX6+MO3	{MUX6,MO3}	{MUX6,MUX6,MO3,MO3}	{MUX6,MO3…}
(4)(8)MUX6+EP7	{MUX6,EP7}	{MUX6,MUX6,EP7,EP7}	{MUX6,EP7…}

In a L-bit MUX problem, environment gives inputs Boolean strings of fixed length L where L denote as $L = k + 2^k$, and the outputs will be a single bit binary output of 0 or 1. The former k bits are called address bit that indexes the remaining 2^k bits, and returns the indexed bit as output. In L-bit MO problems, the output depends on the number of one's quantity. If the number of ones is greater than the number of zeros, the problem instance is of class one, otherwise class zero. In L-bit EP problems, the output depends on the number of ones in the input instance. If the number of ones is even, the output will be one, and zero otherwise.

4. RESULT AND DISCUSSION

To answer the research questions in the current study. First, for #1 and #5, we found that the XCSSTC can learn more than 200 MUX6 problems at the same learning process. This indicates that if all the problems are solvable by the XCS, it may be possible that the XCSSTC could learn them all in a single learning process if N is set to a sufficient value. However, this was achieved by presenting the problem instances in a consecutive manner. We found that the presentation sequence of the problem instances can do harm on the capability of the XCSSTC in learning multiple problems in a same learning process. And furthermore, this harm cannot be overcome by increasing the value of N. Second, for #2 and #6, the results shown that the XCSSTC can learn multiple XCS-solvable problems in a same problem domain but different scales in a same learning process. Third, for #3 and #7, the results shown that the XCSSTC can learn multiple XCS-solvable problems in different problem domains in a same learning process. Fourth, for #4 and #8, that are, for a combination of XCS-solvable and XCS-unsolvable problems, the XCSSTC can learn the XCS-solvable part. The XCSSTC did well in the MUX6 problem but cannot learn the EP7 problem. This could be explained by that the action set pressure [6] favors the classifiers that are more general. When the XCSSTC was learning the EP7 problem instances, the classifiers that are empty in HT when be in the [A] every time and have more opportunity to reproduce. Since the niche size of the EP7 accurate classifiers is small [5] compare to the niche size of the MUX6 accurate classifiers, the EP7 accurate classifiers have less chance to be in [A] than those MUX6 accurate classifiers and hence have less chance to reproduce and larger chance to be eliminated. In the result, the learned EP7 accurate classifiers have greater chance to be eliminated than the learned MUX6 classifiers. Each time when the cycle came to the EP7 problem instances the XCSSTC should re-learn all the accurate rules. We show #5~#8 in the Figure 1.

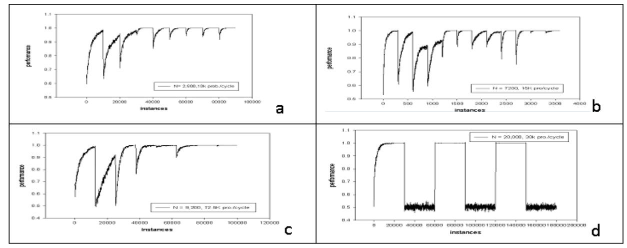

Figure 1. The Performance of the XCSSTC for the (a) $\{MUX6_4\}$ (b) {MUX6, MUX6, MUX11, MUX11} (c) {MUX6, MO3} (d) {MUX6, EP7}.

5. CONCLUSIONS

Our study and previous studies, however, complement each other well, for each emphasizes a different aspect of presentation of XCS. Previous studies try to make different presentation to fit each domain problems, and we make it as a basis to establish a classifier system that can dynamically learn multiple problems at same time to adapt all kind of problems as possible. Furthermore, the behavior and weakness of the XCSSTC in learning various types of combination of problems were also reported... We believe that the proposed approach can lead to building intelligent systems that can dynamically learn multiple problems and also automatically adjust / replace its input attributes and learn to perform complex tasks in larger scale.

6. ACKNOWLEDGMENTS

This work was fully supported by Taiwan Ministry of Science and Technology under grant numbers MOST 103-2221-E-009-139. This work was also supported in part by the "Aim for the Top University Plan" of the National Chiao Tung University and Ministry of Education, Taiwan, R.O.C..

7. REFERENCES

[1] Wilson, S. W. 2000. Get Real! XCS with Continuous-Valued Inputs. *Learning Classifier Systems*, 18132000, 209-219.

[2] Lanzi, P. L. 1999. Extending the Representation of Classifier Conditions Part I: From Binary to Messy Coding. In *Proceedings of the Proceedings of the* Genetic and Evolutionary Computation Conference *computation conference*.

[3] Lanzi, P. L. and Perrucci 1999. A. Extending the Representation of Classifier Conditions Part II: from messy coding to S-Expressions. In *Proceedings of the Proceedings of the genetic and evolutionary computation conference* (1999).

[4] Horowitz, E., Sahni, S. and Mehta, D. 2006. *Fundamentals of Data Structures in C++*. Silicon Press. .

[5] Iqbal, M. 2014. *Improving the Scalability of XCS-Based Learning Classifier Systems*. Dissertation, Victoria University of Wellington.

[6] Butz, M. V., Kovacs, T., Lanzi, P. L. and Wilson, S. W. 2004. Toward a theory of generalization and learning in XCS. *Evolutionary Computation, IEEE Transactions on*, 8, 1, 28-46.

[7] Chen, L.-Y., Lee, P.-M., and Hsiao, T.-C. 2015, "Dynamically Adding Sensors to the XCS in Multistep Problems: a Sensor Tagging Approach", *Proceedings of the Seventeenth International Conference on Genetic and Evolutionary Computation Conference (GECCO2015) Companion, In Press*

[8] Chen, L.-Y., Lee, P.-M., and Hsiao, T.-C. 2015, "A Sensor Tagging Approach For Reusing Building Blocks of Knowledge in Learning Classifier Systems", *IEEE Congress on Evolutionary Computation (CEC2015)*, *In Press*

[9] Chen, L.-Y., Lee, P.-M., and Hsiao, T.-C. 2015, "A Novel Representation of Classifier Conditions Named Sensory Tag for the XCS in Multistep Problems", *Proceedings of the Seventeenth International Conference on Genetic and Evolutionary Computation Conference (GECCO2015)* Companion, *In Press*