Solving Categorization Problem using Two Models of Machine Learning

Lubna Zaghlul Bashir[1] , Nada Mahdi[2]
[1,2]Department of Building and Construction Engineering, University of Technology, Baghdad
e-mail: lubna_zaghlul@yahoo.com

Received : 1/4/2012
Accepted: 1/4/2013

Abstract- The ability to recognize quickly and accurately which we encounter is fundamental to normal intelligent human behavior. However, how the learning of categories which objects in the world fit into takes place is still an unanswered question. One thing is certain though; much of the learning that takes place allows humans to cope with the changing they encounter. One of the most important aspects of human intelligence is its flexibility which has allowed humans to prosper in a dynamic world. Humans do not suffer from the ills of old fashioned hard rule based artificial intelligence. The study tested six cubes. The vertices of the cubes represent individual stimuli constructed from three binary dimensions. The dimension of the stimuli can be assumed to correspond to shape (square vs. circle), color (black vs. white), and size (large vs. small). Four stimuli belonged to one category and the other four to a different category. These constraints result in six problem types, which are illustrated by the six cubes. The circle vertices represent stimuli that belong to category A, and the square vertices represent stimuli that belong to category B. The faces of the cubes represent a constant value across one of the three dimensions that define the stimuli. This work presents experiments with two different classifier systems: learning when fitness is based upon strength and specificity, and learning when fitness is based on strength alone. The system is implemented using Pascal programming language. Results show lower performance of the system when depending on strength alone. By contrast, the run with strength and specificity allows a fast desired output.

Key words:- Strength, Specificity, Fitness, Category, Learning Classifiers System.

1. Genetic Based Machine Learning

Genetic based machine learning considers any system that uses some abstract adoption of Darwinian evolutionary theory, normally in terms of genetic algorithm, in its learning. A genetic algorithm is a search algorithm often used on its own (e.g. for function optimization) but also inside another system as a search engine. The systems are known as Classifier systems. Popularized by Holland, these are "soft rule based" systems. Classifier systems can be used as a universal programming language and are therefore computationally complete. In this sense they are equivalent to other artificial intelligence systems such as production systems and connectionist networks [1], [2] stated the answer to what the term "brittleness" in expert systems was "Induction". Therefore if past experience was to guide future action then a system must learn to discover regularities or reoccurrence within the input. The categories induced by a system must be broad enough to allow inclusion of the likely possibilities, but also specific enough to distinguish those situations requiring different behavior. All rules are of a condition action form and are known as classifiers. The condition part specifies the set of messages for which the rule is active, and the action part specifies a message that is posted by an active rule. Messages are kept to standard lengths allowing conditions to be defined in the same standard length, making the generality of a condition simple to set. Also, rules can be tied together into various networks by the use of tagging. The simple component rules maintain a value that reflects their usefulness within the system. These values are known as the rule's strength. A learning algorithm adjusts rule strengths and as the classifier system learns strength reflects a rule's usefulness over time. In classifier systems the standard approach is to use reinforcement learning scheme though a supervised learning scheme can be adopted. "Reinforcement learning is learning what to do – how to map situations to actions – so as to maximize a numerical reward signal" [3], [4].

2. The Classifier System

The classifier system consists of four major structural components

- A finite population [P] of condition/action rules or classifiers.
- A message list.
- An input interface consisting of a set of environmental feature detectors.
- An output interface for affecting the environment.

Environmental payoff is defined within the problem structure. The system uses three algorithms:

- A performance algorithm called rule and message system.
- A reinforcement learning algorithm called the bucket brigade algorithm.
- A rule discovery algorithm called genetic algorithm **[**1], [2]. Fig.1 illustrates the LCS.

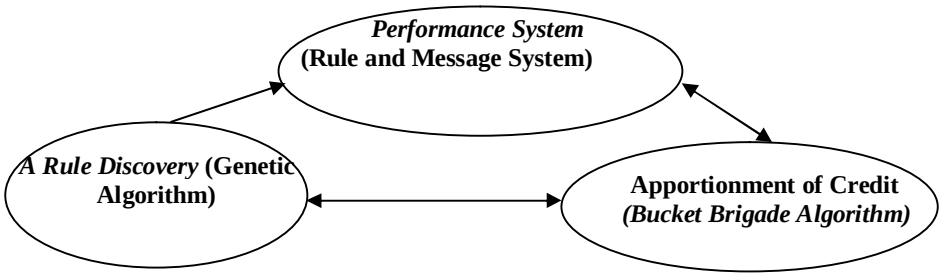

Figure1.The Learning Classifier System (LCS)

2.1. Representation: Classifiers and Messages

The basic elements of all classifier systems are classifiers and messages. All messages are strings of length L from a binary alphabet {0, 1}. All classifiers have the form: $t_1, t_2, \ldots t_r / a$ .The " / " indicates a separation between condition and action. The condition consists of r individual taxa, t_i. The action, a, is a message. Each taxon is a string of length L from a ternary alphabet {0, 1, #}. The condition is satisfied if and only if every t_i matches some message currently on the message list. The individual t_i matches a message if and only if for every 0 or 1 in t_i, the same value occurs in the same position of the one in the message. The "#" functions as a "don't care" symbol in a taxon and matches unconditionally. The # "s confer generality on the taxon, because distinct messages can be matched by a taxon containing n #'s. Thus the classifiers themselves have a quantifiable amount of generality inherited from the #'s in their condition. The output interface is a predefined mapping of some subset of the message space to environmental actions [5], [6]. Fig.2 illustrated the Performance system.

2.2. Performance and Reinforcement Algorithms

The performance and reinforcement algorithms work one after another within a single run cycle of the classifier system. The run cycle of a Holland classifier system operates in stages: first messages from the input interface are posted to the message list; then classifiers that have conditions satisfied by messages on the list post their own messages to the list (previous time step messages are removed); and finally any messages that would trigger the output interface do so, but with resolution of effecter conflicts if necessary (effecter conflict is resolved by the system listening to the classifier with the higher strength). When external payoff from the environment enters the system the amount is shared by all classifiers with messages currently on the list. A classifier whose message is matched by a classifier on the following cycle becomes "coupled" to that classifier. If a sequence of coupled classifiers that leads to external payoff is repeated enough times, then the environmental reward filters backward via the bucket brigade algorithm reinforcing the sequences early acting members. Eventually the initial scene setting classifiers will receive reward for their part in obtaining environmental reward. In

general in these systems classifiers that match on a cycle do not have an automatic right to post their messages, but they must be successful in participating in a biding scheme to post their message. Apart from this all matching classifiers have the right to post their messages and compete for the bucket brigades payoff flow.

2.3. Discovery component

Due to the performance and learning algorithms, classifiers that lead to more environmental payoff increase in strength and thus increase their own and the system's efficiency. In the discovery component strength plays another distinct role, it influences the generation of new classifiers that are likely to be better at obtaining environmental payoff than existing ones. So, the strength of a classifier is used as a measure of that classifier's fitness (in genetic terms) within the population. The discovery component is basically an implementation of Holland's genetic algorithm (GA) [7], [8]. Under the genetic algorithm instances of the best substrings tend to be combined. Therefore, improvements in substring space leads to improvements in classifier space, which result in improvements in system performance. Although the operation of the

Genetic algorithm may seem subtle and complex the actual implementation within the system is quite simple. Classifiers are selected to parent offspring probabilistically over their strengths from the population [P]. Selected parent classifiers are copied and genetic operators are applied probabilistically to the copies. Finally, the modified copies of the parent classifiers known as offspring are added into the population [P], replacing the same number of low (strength) fitness classifiers. The two primary genetic operators within the algorithm are crossover and mutation. In crossover, a randomly selected segment is exchanged between a pair of classifiers. In mutation, the values (alleles) at one or more positions in a classifier are changed to one of the other possible values, in the case of allele in the taxon it would become either a 0 or 1.

3. Related Work

- John Holland introduced a domain – independent rule=based machine learning complex adaptive scheme; a learning classifier system and laid a comprehensive foundation for genetic based machine learning [9].

- Wilson introduced XCS; a learning classifier system derived from his own ZCS and animate problem [7].

- XCS has successfully used for learning Boolean functions by Kovacs and Wilson in the area of data mining [7], [9].

- Barry and Saxon further tested XCS on the Monk's problems [10].

- Wilson evaluated XCS's performance on the Wisconsin breast cancer data set and Holmes did the same on epidemiological data in the domain of medicine [11], [12].

- Decision – free algorithms are also widely used in data mining application [13], [14].

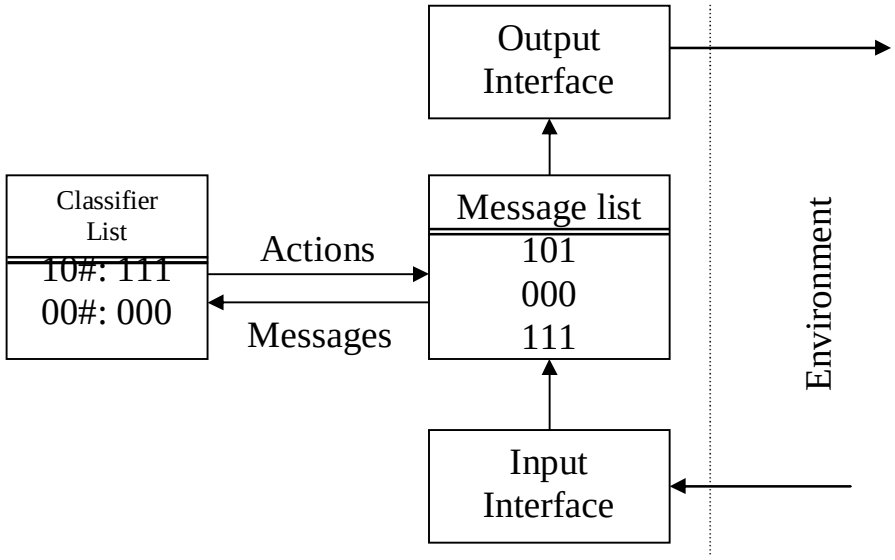

Figure2.The performance system

4. Categorization Problem

Categorization is considered basic to all our intellectual abilities [5]. Categorization "is the process of assigning objects (of whatever kind) to categories (which are collections of objects which are grouped together for some purpose)" [15]. The intelligent behavior of perceptual categorization involves the classification of objects, translated from a physical representation in the external world to an internal representation by a perceptual system. However the ability to effectively acquire knowledge to classify the representations of objects from an agent's external world is no simple problem.

The study tested subjects on six basic types of categorization problem. In each problem there were eight stimuli constructed from three binary dimensions. Four stimuli belonged to one category and the other four to a different category. These constraints result in six problem types, which are illustrated by the six cubes in Fig.3. The vertices of the cubes represent individual stimuli. The circle vertices represent stimuli that belong to category A, and the square vertices represent stimuli that belong to category B. The faces of the cubes represent a constant value across one of the three dimensions that define the stimuli The dimension of the stimuli can be assumed to correspond to shape (square vs. circle), color (black vs. white), and size (large vs. small).Any assignment of four stimuli to each of the two categories can be reflected or rotated into one of the types illustrated in Fig.3.

4.1. The Categorization Problem Cycle

The Categorization Problem consists of two learning classifier systems. The mechanism of the two classifier systems is

the same. First, the system receives a message from environment through detectors into a single message list. This message matches against classifiers in the classifier store to perform match list. Second, an action is held to choose a winning classifier. The action of wining classifier goes forward to the environment as output. At last reinforcement is provided to the system to reward the classifier of good action and punish the classifier of bad action. Fig.4 illustrates Categorization Problem cycle.

4.2. The Performance System for Categorization Problem

The Performance system of the Categorization Problem consists of a message list and classifier store. There is only a single message in the message list that is used to match against the condition part of all classifiers in the classifier store and there is no message to be received in the current cycle until the system produces an action.

Coding (LCS) Conditions

The learning classifier system (LCS) receives 3 – bit messages from environment mapping it to 8 states from 0 to 7 bits only. The three bit is represented as follows:

First bit represents the object shape (1 means the object is circle, 0 means that object is square).

Second bit represents the object color (1 means the object is black, 0 means that object is white).

Third bit represents the object size (1 means the object is small, 0 means that object is large).

Coding (LCS) Actions

LCS has an action consisting of only one bit. LCS action has the form and meaning as follows:

- 1 means the object belongs to category A.
- 0 means the object belongs to category B.

Representation of (LCS)

The Performance system of the *LCS* consists of a message list and classifier store. The classifier list of LCS contains a set of rules called classifiers which represents the knowledge and controller of the system at execution time. The condition part of classifier consists of (3 bit), and action part consists of (1 bit). The size of classifier store for *LCS* will be (8) Rules and all classifiers have the same strength value at the beginning.

Example

The representation of the rule "If object shape is square AND object color is black AND object size is large THEN the object belongs to category A ":

shape	color	size	/	category A
1	1	1	/	1

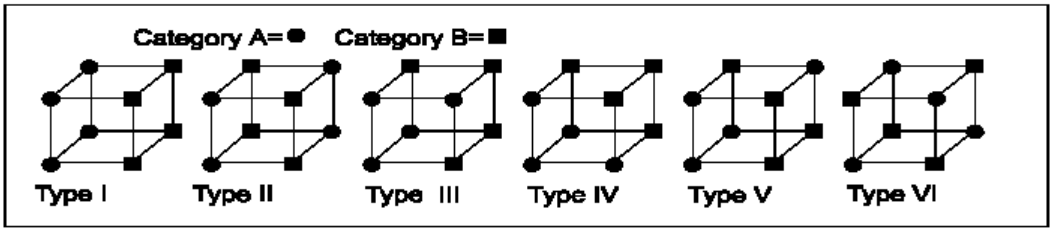

Figure 3. The Six Type of Categorization Problem

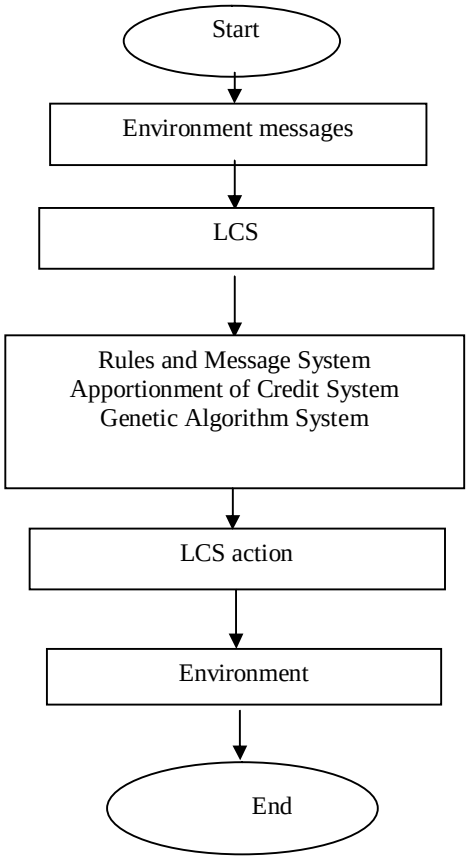

Figure4. Categorization Problem cycle

4.3. Apportionment of Credit (AOC) for Categorization Problem

The procedure (AOC) in the Categorization Problem calls three routines: auction, clearing house and tax collector.

First procedure, the function auction holds a noisy auction to select a winning classifier from the set of matched classifiers. The auction cycle through the matched classifiers successively calculates each classifiers base (bid) and its effective bid ((Ebid) as illustrated in equations (1),(2),(3) and (4) respectively. The function auction keeps track of the classifier index with highest effective bid and returns this value upon relinquishing control to procedure (AOC).

Formally for Categorization Problem, a classifier's bid is defined as a product of C_{bid} and a linear function of classifier's specificity, and classifier strength [16].

$$Bid_i = C_{bid} * f(SP) * S_i \quad (1)$$

$$f(SP) = bid_1 + bid_2 * SP \quad (2)$$

Where: C_{bid} is the bid coefficient, usually C_{bid} a constant less than 1. Specificity (SP) = number of non # /Total Length of condition part. bid_1, bid_2 are input parameters. S Is strength, i is classifier index.

The effective bid (EB) for each matched classifier is the sum of its deterministic bid and a noise term:

$$EB_i = Bid_i + N(\sigma_{bid}) \quad (3)$$

Where, the noise N is a function of the specified bidding noise standard deviation σ_{bid}.

The winning classifiers place their messages on the message list and their strengths are reduced by the amount of their bids. Typically, if $S_C(t)$ denotes the strength of a winning classifier, C at time t and $B_C(t)$ denotes its bid at the same time t, then its strength at time $t+1$ is:

$$S_C(t+1) = S_C(t) - B_C(t) \quad (4)$$

Then after procedure clearinghouse is invoked to reconcile payments. The current winners' strength is simply decreased by the amount of its bid value for the Categorization Problem. Bucket brigade algorithm flag is usually set to false because reward is available at every time step and because there is no relationship between successive signals.

For Categorization Problem if C' sent the message matched by C above, and then the strength of C' at time $t+1$ is:

$$S_{C'}(t+1) = S_{C'}(t) + B_C(t) \quad (5)$$

The last routine called by the AOC procedure is tax collector. To discourage nonproductive classifiers, two different types of tax are collected from classifiers; an existence tax and bid tax. The existence tax is assessed and collected from all classifiers at a tax rate specified in the real value population variable life tax. The bid tax is assessed and collected from all classifiers that bid in the last auction; this tax rate is specified by the real variable bid tax. Many schemes are available; a tax was simply collected proportional to the classifier strength:

$$T_i = C_{tax} * S_i \quad (6)$$

Where: T is tax**,** C_{tax} is coefficient of tax**,** S is strength and i is classifier index**.**

To implement a well-defined procedure the auction and payment scheme was detailed. Classifiers make ***bids*** (B_i) during the auction. Winning classifiers turn over their bids to the clearinghouse as ***payments*** (P_i)**.** A classifier may also have ***receipts*** (R_i) from its previous message- sending activity or from environmental reward. In addition to bids and receipts, a classifier may be subject to one or more ***taxes*** (T_i). Taken together, the equation governing the depletion or accretion of the classifier strength is as follows [8]:

$$S_i(t+1) = S_i(t) - P_i(t) - T_i(t) + R_i(t) \quad (7)$$

The apportionment of credit equation recasts into a more useful form where all payments and taxes have been replaced by their strength equivalent [16], [17].

$$S(t+1) = S(t) - C_{bid} * S(t) - C_{tax} * S(t) + R(t) \quad (8)$$

4.4. Rule Discovery for Categorization Problem

GA is a way of injecting new possibly better rules into the system (Categorization Problem). New rules are created by the rule discovery process, ***Reproduction, crossover, and mutation.*** These rules are then placed in the population and processed by the auction, payment and reinforcement learning to properly evaluate their role in the system. [17], [18].

4.5. The Reinforcement Learning of Categorization Problem.

Reinforcement algorithm is preferable to be provided by a human trainer who observes the performance of the system and provides positive or negative reinforcement according to the action to be taken.

4.6. Comparison between Learning Depended Strength Alone vs. Strength and Specificity

The categorizing tasks are easily expressed in terms of binary strings to learn. The system was run over the categorizing tasks consisting of 100 trials, with the parameters illustrated in Fig.6.

Learning using classifier system depended strength alone for evaluation we start from the same initial population, achieve this by using **bid1=ebid1=1 and bid2=ebid2=0**.insted of learning using strength and specificity bid1**=ebid1=0.250 and bid2=ebid2=0.125**, as shown in Table1.

The performance of the system to solve cubic problem after 100 trials illustrates the slowness of learning task to solve the cubic problem when compared to learning depended strength and specificity to solve the same problem. As shown in Fig.8, the environmental message won after 94 iterations when using learning depended strength alone. The same message won after16 iterations as in Fig.7 when using learning depended strength and specificity. Table 2 illustrated numbers of iterations for the six quips when we use strength only or use strength &specificity.

In addition, without using specificity the default rule [##1:1] wins against the perfect rule [101:1] when it should not. In the run with specificity bid the structure depends on enough difference existing between specific and general rule bids to allow consistent selection of exception

rules when they are matched. Table 3 illustrated winning rules.

The run of the program using learning with strength only is unable to perform as well as the run using learning with combination of strength and specificity. The time need to match signal is shorter when using learning with strength and specificity, and we are more likely to achieve good result quickly. The performance of the system after 100 iterations for qubic-1 is illustrated in Table 4 and data chart illustrated in Fig.5.

Table1. Bid Structure.

	Bid1	Ebid1	Bid2	Ebid2
Learning using strength &specificity	0.250	0.250	0.125	0.125
Learning using strength only	1	1	0	0

Table2. Numbers of Iterations for The Six Quips.

	Number of iterations using strength &specificity	Number of iterations using strength only
Cubic-1	16	94
Cubic-2	15	97
Cubic-3	15	100
Cubic-4	17	98
Cubic-5	14	92
Cubic-6	16	97

Table3. Wining Rule for The Six Quips.

	Wining rule using strength &specificity	Wining rule using strength only
Cubic-1	101:1	1##:1
Cubic-2	101:0	##0:0
Cubic-3	101:1	##1:1
Cubic-4	101:0	1##:1
Cubic-5	101:1	#0#:1
Cubic-6	101:0	0#1:0

Table4. The System Performance After 100 Iterations

No of	**P(correct) using**	**P(correct) using**
Iterations	strength & specificity	strength only
10	0.8	0.6
20	0.88	0.65
30	0.9	0.7
40	0.95	0.72
50	0.96	0.75
60	1	0.77
70	1	0.78
80	1	0.78
90	1	0.8
100	1	0.81

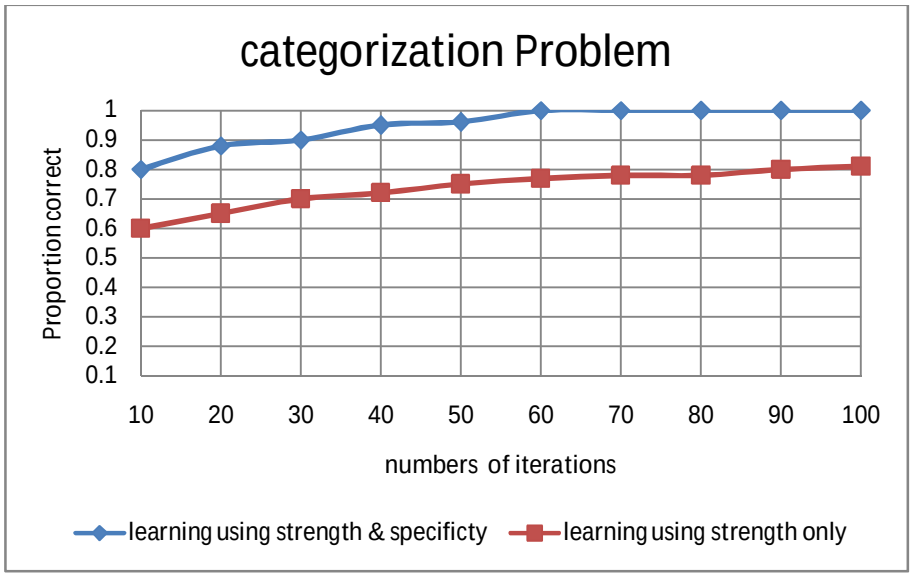

Figure 5. The System Performance After 100 Iterations

5. Conclusions

- This paper shows how two different classifier systems are both able to demonstrate similar learning performance over a set of classification tasks.

- The results show that the learning depended strength alone is slower than learning depended strength and specificity when both of them start with the same initial population.

- Learning using fitness depended strength alone is slower to recover because its general rule wins against perfect rules when it should not. This lowers the performance of the system. By contrast, in the run with specificity, enough difference exists between the specific and general rule to allow consistent selection of exception rules when they match.

- Genetic based machine learning can be used as a model of learning.

- In the long run for big knowledge based systems, learning will turn out to be more efficient than programming.

- Experimentally comparing between a learning algorithm with a combination of strength and specificity produces better results than either strength.

- Experiment shows that the time needed to perform the task when using learning with strength and specificity is less than the time needed when using learning with only strength.

- Experimentally we obtain more accuracy of the signal when using learning with strength and specificity.

The simplest category structure is a Type 1 problem, where only information from a single dimension is relevant to solve the classification problem.

==========

A SIMPLE CLASSIFIER SYSTEM FOR CUBIC-1 USING STRENGHT & SPECIFICITY

population parameters
---
number of classifiers = 24
number of positions = 3
number of action = 1
bid coefficient = 0.1000
bid spread = 0.0750
bidding = 0.0100
existence tax = 0.0200
generality probability = 0.5000
bid specificity base = 0.2500
bid specificity mult. = 0.1250
edid specificity base = 0.2500
ebid specificity mult. = 0.1250

environmental parameters
---
total number of signal = 3
apportionment of credit parameters
---
bucket brigade flag = false

reinforcement parameters
---
reinforcement reward = 10.0
Timekeeper parameters

Initial iteration = 0
Initial block = 0
Report period = 1
Console report period = 1
Plot report period = 1
Genetic algorithm period = 1
Genetic Algorithm Parameters

Proportion to select/gen = 0.6000
Number to select = 7
Mutation probability = 0.0200
Crossover probability = 1.0000
Crowding factor = 3
Crowding subpopulation = 3
\snapshot report
[block: iteration] - [0:0]
current status
---
signal = 000
Decoded signal = 0
desired output = 0
classifier output = 0
environmental message: 000

no.	strength	bid	ebid	M	classifier
1	10.00	0.00	0.00		0##:[0]
2	10.00	0.00	0.00		0##:[1]
3	10.00	0.00	0.00		1##:[0]
4	10.00	0.00	0.00		1##:[1]
5	10.00	0.00	0.00		##0:[0]
6	10.00	0.00	0.00		##0:[1]
7	10.00	0.00	0.00		##1:[0]
8	10.00	0.00	0.00		##1:[1]
9	10.00	0.00	0.00		#0#:[0]
10	10.00	0.00	0.00		#0#:[1]
11	10.00	0.00	0.00		#1#:[0]
12	10.00	0.00	0.00		#1#:[1]
13	10.00	0.00	0.00		11#:[0]
14	10.00	0.00	0.00		11#:[1]
15	10.00	0.00	0.00		#11:[1]
16	10.00	0.00	0.00		00#:[1]
17	10.00	0.00	0.00		#00:[0]
18	10.00	0.00	0.00		##0:[0]
19	10.00	0.00	0.00		1#1:[1]
20	10.00	0.00	0.00		0#0:[0]
21	10.00	0.00	0.00		1#0:[0]
22	10.00	0.00	0.00		0#1:[1]
23	10.00	0.00	0.00		###:[0]
24	10.00	0.00	0.00		###:[1]

new winner[1] : old winner[1]

Figure6. Initial report for Cubic -1

[block: iteration] - [0:16]
current status
signal = 101
Decoded signal = 5
desired output = 1
classifier output = 1
environmental message: 101

no.	strength	bid	ebid	M	classifier
1	75.82	3.91	4.00	x	#01:[1]
2	102.95	0.00	0.00		111:[1]
3	92.01	0.00	0.00		111:[1]
4	92.63	5.97	6.03	x	101:[1]
5	98.97	6.13	6.08	x	101:[1]
6	88.09	0.00	0.00		111:[1]
7	101.08	5.21	5.25	x	1#1:[1]
8	87.81	4.53	4.42	x	1#1:[1]
9	99.83	5.15	5.19	x	1#1:[1]
10	87.81	4.53	4.56	x	1#1:[1]
11	92.01	0.00	0.00		1#0:[1]
12	99.12	5.11	5.08	x	1#1:[1]
13	87.19	5.62	5.59	x	101:[1]
14	101.08	5.21	5.16	x	1#1:[1]
15	87.96	0.00	0.00		1#0:[1]
16	114.44	5.90	5.85	x	1#1:[1]
17	102.13	0.00	0.00		111:[1]
18	81.00	3.13	3.07	x	1##:[1]
19	99.12	5.11	5.05	x	1#1:[1]
20	92.63	4.77	4.69	x	1#1:[1]
21	96.06	0.00	0.00		111:[1]
22	96.08	0.00	0.00		1#0:[1]
23	101.08	5.21	5.12	x	1#1:[1]
24	91.60	0.00	0.00		111:[1]

new winner[5] : old winner[16]

Figure7. Last report for Cubic -1using learning depend strength and specificity

==========
A SIMPLE CLASSIFIER SYSTEM FOR CUBIC-1 USING STRENGHT
==========
snapshot report
[block: iteration] - [0:94]
current status
signal = 101
Decoded signal = 5
desired output = 1
classifier output = 1
environmental message: 101

no.	strength	bid	ebid	M	classifier
1	76.34	7.62	7.53	x	**1**##:[1]
2	52.14	5.37	5.41	x	0#1:[0]
3	61.90	6.38	6.37	x	0##:[1]
4	61.18	0.00	0.00		01#:[1]
5	50.51	0.00	0.00		010:[0]
6	55.61	0.00	0.00		010:[0]
7	60.89	6.28	6.21	x	0##:[1]
8	64.80	6.68	6.66	x	0#1:[1]
9	57.41	5.92	5.97	x	0##:[1]
10	49.94	0.00	0.00		1##:[0]
11	53.93	5.56	5.59	x	0#1:[1]
12	56.72	0.00	0.00		01#:[1]
13	64.80	6.68	6.68	x	0##:[1]
14	62.96	6.49	6.47	x	0##:[1]
15	57.95	5.97	5.96	x	0#1:[1]
16	61.53	6.34	6.52	x	0##:[0]
17	65.59	0.00	0.00		01#:[1]
18	56.51	5.83	5.84	x	0##:[1]
19	60.10	6.20	6.38	x	0##:[0]
20	60.72	0.00	0.00		01#:[1]
21	61.90	6.38	6.37	x	0##:[1]
22	55.05	5.67	5.73	x	00#:[0]
23	61.53	6.34	6.35	x	0##:[1]
24	60.89	6.28	6.25	x	0##:[1]

new winner[1] : old winner[1]

Figure8. Last report for Cubic -1using learning depend strength only

References

[1] J. H. Holand , "Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule based systems." In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (eds), *Machine learning: an artificial intelligence approach* (Vol. 2). Los Altos, CA: Morgan Kaufmann, 1986.

[2] S. Sen, "Modeling human categorization by a simple classifier system", 1996.

[3] R. Hartley, "Genetics based machine learning as a model of perceptual category learning in humans". M. Sc Thesis. Computer Science Department, University of Birmingham, UK, 1998.

[4] R. S. Sutton, & A. G. Barto, "Reinforcement learning: An introduction". London: MIT Press, 1998

[5] W.K. Estes, "Classification and cognition". New York: Oxford University Press, 1994.

[6] A.R. Hartley, "CLASSIFIER SYSTEMS: Accuracy-based fitness allows similar performance to humans in static and dynamic classification environments". The University of Birmingham School of Computer Science Edgbaston Birmingham, B15 2TT United Kingdom Email arh@cs.bham.ac.uk,2000.

[7] S. W. Wilson, "Classifier systems and the animate problem". *Machine Learning*. 2, 199-228, 1987.

[8] L. Smuel, S. Olivier, "A Comparison between ATNoSFERES and Learning Classifier Systems on non-Markov problems", Institut des Systèmes Intelligents et de Robotique, CNRS FRE 2507Université Pierre et Marie Curie - Paris 64 place JussieuF-75 252 Paris Cedex 05FranceAb, 2008.

[9] D E. Goldberg, "Genetic Algorithm in Search, Optimization, and Machine Learning", Addison Wesley Longmont, International Student Edition, 1989.

[10] S.Saxon and A. Barry,"XCS and Monk's problems", proceedings of the genetic and evolutionary computation conference, volume, page 809, Orlando, Florida, USA, 13-17. Morgan Kaufmann, 1999

[11] S.W. Wilson, "Mining obligue data with XCS lecture notes in computer science, 2001.

[12] J. H. Holmess, "learning Classifier Systems applied to knowledge discovery in clinical research database" in learning classifier systems from foundations to applications, page 243 262,landon UK.spriger-verlag,2000.

[13] I.H. Witten and E. Frank, "data mining :practical machine learning tools with java implementations", Sanfran Cisco, USA,2000.

[14] Sr. Anil and J.L Sushil, ''Better Personalization using Learning Classifier Systems" department of computer science and engineering, the university of Nevada, united states of America,2005.

[15] K. Lamberts, "Process models of categorization", In K. Lamberts & D. Shanks (Eds.), Knowledge, concepts and catagories, London: UCL Press, 1997.

[16] A. Kharmandar, A. Naeimi, A. M. Alizadeh, S. Jafari, S. Chavoshi, "Soccer Simulation 2DTeam Description Proposal for Robocup, Payame Noor University, Iran,2011.

[17] B. Jason, "Learning Classifier Systems", Technical Report 070514AComplex Intelligent Systems Laboratory, Centre for Information Technology Research, Faculty of Information and Communication Technologies, Swinburne University of Technology Melbourne, Australiajbrownlee@ict.swin.edu.au, 2007

[18] E. Anders, "Evolution of Meta-parameters in Reinforcement Learning" Master's Thesis in Computer Science, at the School of Computer Science and Engineering, Royal Institute of Technology, Stockholm, Sweden,2002.