Using Classifier System with / without Genetic Algorithm in Robotics Behaviors

Lubna Zaghlul Bashir
Department of Building and Construction Engineering, University of Technology
(Baghdad, Iraq)
Email: lubna_zaghlul {at} yahoo.com

ABSTRACT--- ***A learning classifier system is one of the methods for applying a genetic-based approach to machine learning applications. An enhanced version of the system that employs the Bucket-brigade algorithm to reward individuals in a chain of co-operating rules is implemented and assigned the task of learning rules for control robotics behaviors. Illustrates the approach with the example of kicking a moving Ball into a goal (KMB). The KMB (kick a Moving Ball) System built of two-classifier subsystems work together, each classifier system learns a simple behavior, first classifier system, learn simulated robot chase behavior i.e. learn robot to move single step toward moving ball, second classifier system, learn the simulated robot approach behavior i.e. learn robot to kick the ball toward fixed goal, the system as a whole has as its learning goal the coordinate of behaviors. This work examine the performance of the simple classifier system (SCS) on the (KMB) problem, perform two SCS simulations one without the genetic algorithm enabled (GA) and one with the genetic algorithm enabled (GA) results using a classifier system with genetic algorithm show improvement over one without, and furthermore the level of performance has been high enough to rival human accuracy. Also the results without the genetic algorithm show that the apportionment of credit algorithm adjust the strength values of the rules.***

Keywords--- Robotics behavior, Learning Classifier Systems, genetic algorithm

1. INTRODUCTION

A Holland Classifier Learning system is one of the ways of using evolutionary methodology for machine learning applications. These systems are a class of rule-based, message processing systems, [**1, 2, 3].** Rules in these systems are known as classifiers because they are mainly used to classify messages into general sets. Many rules could be active simultaneously since the only action of an active classifier is to post a message onto a message list. Learning in classifier systems is achieved by two mechanisms: **Bucket-brigade** and **Genetic Algorithms, [1].** Bucket-brigade is designed for allocating credit to classifiers or rules according to their usefulness in attaining systems goals. It is also designed as a fitness function; rating each classifier in the system. Genetic Algorithms are used to search for new plausible classifiers or rules by recombining good classifiers to produce new ones.

2. LEARNING CLASSIFIER SYSTEMS

Classifier system is used as a machine learning system that learns syntactically simple string rules (called classifier) to guide its performance in an arbitrary environment .a classifier system consist of three main components: 1.Performance System (Rule and Message System). 2. Apportionment of Credit System (Bucket Brigade Algorithm).3.Genetic algorithm (Rule Discovery). [4, 5, 6]

2.1 The Performance System.

The performance system is composed of:

1. Classifier List

The classifier list is the system's long term memory. It is made up of a population of classifiers. A classifier is made up of one or more conditions (known as the **condition part**) and one action (called the **action part**). The condition part specifies the set of messages to which a classifier is sensitive, and the action part indicates the message it will broadcast or send out when its condition part is satisfied. Thus, a classifier list consists of one or more classifiers of the form: $\mathbf{C_1, C_2, ..., C_n/a}$

Where C_1, C2,….C_n {n>=1}are the conditions making up the condition part and 'a' is the action part, conditions are connected by AND operator. Each C_i is a string of fixed length K over a fixed alphabet. In most practical systems, the string is defined over three alphabets: {1,0, #}. The '#' is a don't care (wild card) symbol that can match any of the chosen symbols. A classifier posts one or more messages onto the message list when it is activated. The action part of a classifier is used to form the message it sends out when it is activated. It is also a string of fixed length K defined over the alphabet {1, 0}.**[4,7].**

2. Message List

The message list acts as the system's short-term memory and as the medium for communication between classifiers, and the output interface. It is made up of external messages (input observations) and internal messages (messages from classifiers). A message is represented by a string of fixed length K (same length as that for a condition) over the same set of alphabets {1, 0} as the action. **[7, 8, 9].**

3. Input Interface (Detectors)

This receives the input messages from the environment and transforms them into fixed length strings to be placed on the message list **[8].**

4. Output Interface (Effectors)

Messages placed on the message list by classifiers are processed through the output interface in order to communicate with the system's environment**. [8]**

2.2 Apportionment of Credit System (Bucket-Brigade Algorithm)

The bucket-brigade algorithm is designed to solve the credit assignment problem for classifier systems and to determine the worth of each classifier. The credit assignment problem is that of deciding which of a set of early active classifiers should receive credit for setting the stage for later successful actions. To this end, a numerical quantity (called strength) is assigned to each classifier. This strength is adjusted continually by the algorithm to reflect the classifier's past usefulness. Each classifier whose condition part is satisfied by one or more messages makes a bid to post a message onto the message list. Only the highest bidders are allowed to become active and hence post messages. A classifier's bid depends on its strength and the specificity of its condition. The specificity measures the relevance of a classifier's condition to a particular message. Formally, a classifier's bid is defined as:**Bid = C_{bid}*strength*specificity.** Where **specificity** = number of non-#s/Total Length of condition part. **C_{bid}** = a constant less than 1. The winning classifiers place their messages on the message list and their strengths are reduced by the amount of their bids. **[10, 11].**

2.3 Genetic Algorithm (Rule Discovery).

There exist three major classes of genetic operators:

- ***Selection:*** this operator selects chromosomes in the population for reproduction. The fitter the chromosome, the more times it is likely to be selected to reproduce.
- ***Crossover:*** this operator randomly chooses a locus and exchanges the subsequences before and after that locus between two chromosomes to create two offspring. For example, the strings **10000100** and **11111111** could be crossed over after the third locus in each to produce the two offspring **10011111** and **11100100**.
- ***Mutation:*** this operator randomly flips some of the bits in a chromosome. For example, the string **00000100** might be mutated in its second position to yield **01000100.**The Simplified architecture of learning classifier system illustrate in **Figure.1. [5, 12, 13].**

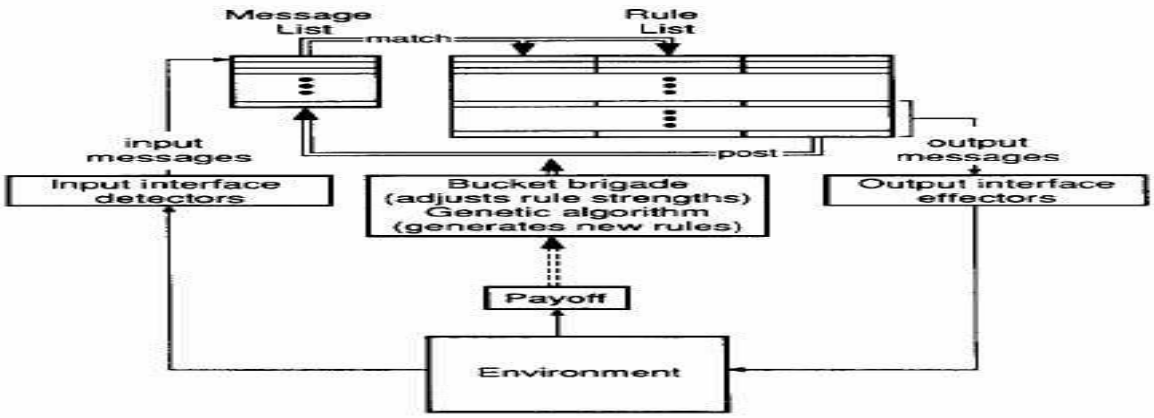

Figure.1: Simplified Architecture of Learning Classifier System

3. KICK A MOVING BALL: A CASE STUDY

In this study, the system called ***Kick a Moving Ball (KMB)*** uses two classifier systems to implement controller for simulated robot, which has flat architecture. In the system the robot should learn complex behavior consisting of two basic behaviors: Chase behavior and Approach behavior. Two classifier systems were used to perform complex behavior. First classifier called (LCS-Position) learns the robot to move one step toward the moving ball (chase behavior).second classifier called (LCS-Kick) learn the robot kick the ball toward goal (approach behavior)when there is

no predator such as existing man . The objects in environment are as follows: Moving robot, moving ball, and fixed position represented by goal. The environment illustrated in **Figure.2.**

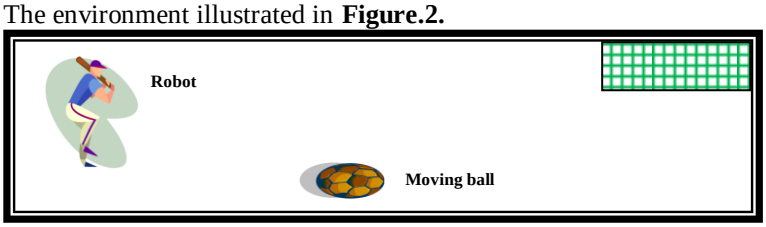

Figure.2: The Kick Moving Ball Environment.

3.1 (KMB) System Structure

The KMB system is built of two learning classifier systems. Organized in flat architecture, interacting together to perform complex behavior, consist of two **Learning Classifier Systems** (LCS-Position) and (LCS-Kick).the KMB structure is shown in **Figure.3**.

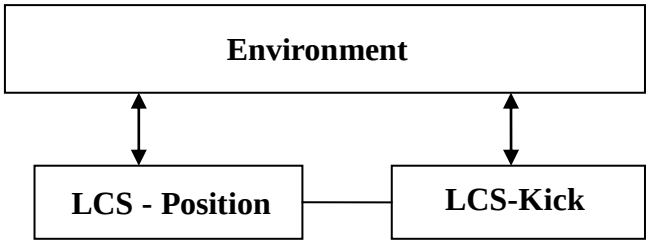

Figure.3: KMB System Structure.

3.2. (LCS-Position) Development

LCS-Position is used to learn robot move single step toward moving ball. The movement capability is completely symmetric along the two axis .The direction of movement is illustrated in

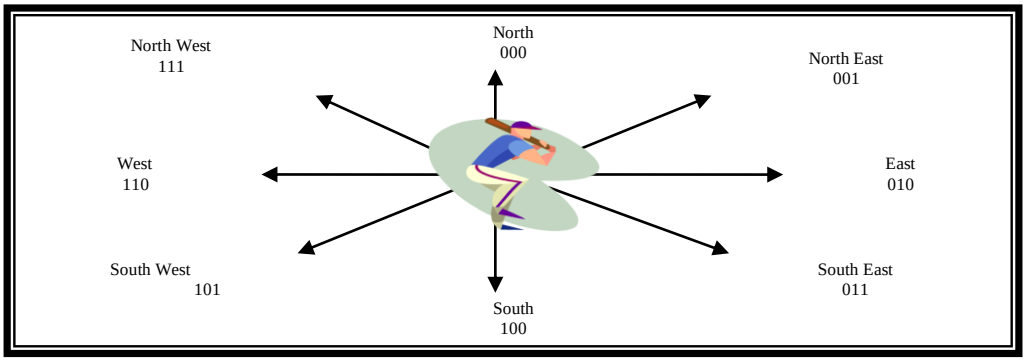

Figure.4: Direction of Robot Movement

3.2.1 Coding (LCS – Position) Conditions

The length of message, which LCS – Position is received is always, 3 – bit environment message mapping it to eight states from 0 to 7 of three bit only. The meaning of three bit in input message of LCS – Position determine relative position of ball from robot. The form and meaning of three bit LCS – Position is shown in **Table 1**.

Table 1: Form and meaning of LCS – Position Conditions

The message	Its meaning
0 0 0	Relative position of ball from robot is to north
0 0 1	Relative position of ball from robot is to north -east
0 1 0	Relative position of ball from robot is to east
0 1 1	Relative position of ball from robot is to south -east

1 0 0	Relative position of ball from robot is to south
1 0 1	Relative position of ball from robot is to south-west
1 1 0	Relative position of ball from robot is to west
1 1 1	Relative position of ball from robot is to north - west

3.2.2 Coding (LCS – Position) Actions

The desired action should be the same as system input message. Therefore, we have eight actions. Action has the form and meaning as in **Table2**.

Table 2: Form and meaning of LCS – Position actions

The action	Its meaning
0 0 0	Means robot move to the north
0 0 1	Means robot move to the north - east
0 1 0	Means robot move to the east
0 1 1	Means robot move to the south - east
1 0 0	Means robot move to the south
1 0 1	Means robot move to the south – west
1 1 0	Means robot move to the to west
1 1 1	Means robot move to the north – west

3.2.3 Representation of (LCS – Position)

LCS – Position consists of a condition part of (3bit) representing the position of ball in the environment and form action part of (3 bit) representing action to be done in the environment, the size of classifier store for LCS – Position will be (8) rules.

Example: The representation of the rule "if a relative position of a ball to be sensed from simulated robot is to the north then the action to be taken by simulated robot is moving to the north, and so on.

Position of ball from the robot / direction moving of robot

0 0 0 / 0 0 0

3.3 (LCS – Kick) development

LCS – Kick is used to learn simulated robot Kick the ball toward goal in approach behavior. The movement capability is completely symmetric along the two axes.

3.3.1 Coding (LCS – Kick) Conditions

The length of message, which LCS – Kick is received is always, 3 bit environment message mapping it to eight states from 0 to 7 of three bits only. The meaning of three bit in input message of LCS – Kick determines relative position of the goal from ball. The form and meaning of three bit LCS – Kick is shown in **Table 3**.

Table 3: form and meaning of LCS - Kick Conditions

The message	Its meaning
0 0 0	Relative position of goal from ball is to north
0 0 1	Relative position of goal from ball is to north - east
0 1 0	Relative position of goal from ball is to east
0 1 1	Relative position of goal from ball is to south - east
1 0 0	Relative position of goal from ball is to south
1 0 1	Relative position of goal from ball is to south -west
1 1 0	Relative position of goal from ball is to west
1 1 1	Relative position of goal from ball is to north - west

3.3.2 Coding (LCS – Kick) Actions

The desired action should be the same as the system input message. Therefore we have eight actions. Action has the form and meaning as in **Table.4**

Table 4: Form and meaning of LCS – Kick actions

The action	Its meaning
0 0 0	Means robot kick the ball to the north
0 0 1	Means robot kick the ball to the north - east
0 1 0	Means robot kick the ball to the east
0 1 1	Means robot kick the ball to the south - east
1 0 0	Means robot kick the ball to the south
1 0 1	Means robot kick the ball to the south – west
1 1 0	Means robot kick the ball to the west
1 1 1	Means robot kick the ball to the north – west

3.3.3 Representation of (LCS – Kick)

LCS – Kick consists of a condition part of(3bit) representing the fixed position of goal in the environment and form action part of (3 bit) representing robot action to be done in the environment the size of classifier store for LCS – Kick will be (8) rules.

Example: The representation of the rule "if a relative position of a goal from ball is to the north - west then the action to be taken by robot kick the ball to the north - west, and so on.

Position of goal from the ball / direction moving of ball

1 1 1 / 1 1 1

4. THE PERFORMANCE SYSTEM FOR (KMB) SYSTEM

As the performance system is the heart of the classifier system, the matching procedures are the heart of the performance system. Performance system of the KMB consists of a message list and classifier store. There is only single message in the message list that is used to match against condition part of all classifiers in the classifier store and there is

no message to be received in the current cycle until the system produce an action. The two routines are responsible for matching classifiers to the environment message: match and match classifiers.

The function match performs a match between a single condition and a single message and returns a Boolean true value if match succeeds. The procedure match classifiers match all classifiers against the environment message and construct the match list data structure.

5. APPORTIONMENT OF CREDIT (AOC) FOR (KMB) SYSTEM

The procedure (AOC), in the system (KMB) calls three routines: auction, clearing house and tax collector. In the first procedure, the function auction holds a noisy auction to select a winning classifier from the set of matched classifiers. auction cycle through the matched classifiers , successively calculates each classifiers base (bid) and its effective bid (Ebid) as illustrated in equations: (1),(2),(3),(4) 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 KMB system, a classifier's bid is defined as a product of C_{bid} and a linear function of classifier's specificity, and classifier strength **[7]**.

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

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

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: $S_C(t+1) = S_C(t) - B_C(t)$ **(4)**

There after procedure clearinghouse is invoked to reconcile payments. the current winners strength is simply decreased by the amount of its bid value for the KMB 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 Example 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: $T_i = C_{tax} * S_i$ **(6)**

Where: T is tax, C_{tax} is coefficient of tax, S is strength, i 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 as follows **[3,14]:**

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

The apportionment of credit equation recasts into a more useful form where all payments and taxes have been replaced by their strength equivalent **[7]**.

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

6. Rule Discovery for (KMB) System

GA is a way of injecting new possibly better rules into the system (KMB) 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.

- In the work (KMB) a quantity called the selection proportion were used, where replace that proportion of the population at a given genetic algorithm invocation.
- ***The number of mate pairs to select =proportion select*n classifier*0.5*** **(9)**

In classic implementation of Classifier Systems, the genetic algorithm (i.e. reproduction, crossover and mutation) is called every T_{ga} cycles where T_{ga} is a constant whose optimal value is experimentally determined. The invocation of genetic algorithm learning may be conditioned on particular events such as lack of a match or poor performance. The basic execution cycle (the central loop) as in **Figure. 5.**

```
Procedure GA;
{Coordinate selection, mating, crossover, mutation, & replacement}
Begin with population do
     begin
       Statistics (population);      {get average, max, min, sumstrength}
       For j:=1 to n select do with mating[j] do
           begin
               mate1:=select(population);            {pick mates}
              mate2:=select(population);
Crossover(classifier[mate1],classifier[mate2],child1,child2,
pcrossover, pmutation, sitecross, nposition, ncrossover, nmutation)
mort1:=crowding(child1,population,crawdingfactor,crawdingsubpop);
sumstrength:=sumstrength-classifier[mort1].strength+child1.strength
classifier[mort1]:=child1;
mort2:=crowding(child2,population,crawdingfactor,crawdingsubpop);
sumstrength:=sumstrength-classifier[mort2].strength+child2.strength
classifier[mort2]:=child2;
          end;
end   end;
```

Figure. 5: The Genetic Procedure in LCS

Selection of next Generation

Now searching, not for the single best rule (classifier), but for a well-adapted set of rules. Therefore use the "crowding replacement" algorithm to choose the classifiers that should die to make room for new offspring. (This implies combining the best of the parents and offspring.) Crowding replacement aims to replace a low performing classifier with a similar (potentially better classifier) **[3,7]**.

- The selection process for (KMB) is performed using ***roulette wheel*** selection where each classifier's strength value S is used as its fitness. The sum of the population fitness is calculated .The expected value of an individual is the individual's fitness divided by the average fitness of the population.
- The crossover operator is implemented by taking two-parent string and generating two offspring string. In (KMB) problem crossover is implemented as follows: an integer position k along the string is selected uniformly at random between 1 and the string length less one [1, l-1]. Two new strings are created by swapping characters between position k+1 and l inclusively.
- When a mutation is called it changes the mutated {0, 1,#} character to one of other two with equal probability**.**

7. EXECUTING OF SYSTEM KICK MOVING BALL CODE (KMB)

The whole project was programmed in Pascal language, Executing the Kick Moving Ball code, the system responds by presenting the initial report display in **Appendix A** for (LCS – Kick) .the classifier system run for 100 iterations, termination with the last snapshot report display in **Appendix A** the correct rules have achieved high strength values, by contrast the bad rules have strength and bid value near zero. The classifier system eliminates the bad rule quickly there by achieving near perfect performance. This work examines the performance of the simple classifier system on kick moving ball system starting from a randomly generated set of rules. we perform two SCS simulations, one without the genetic algorithm enabled(no GA) and one with the genetic algorithm enabled (GA) .in this way we are able to separate the learning due to apportionment of credit among the original rules and that due to the injection of new rules by the genetic algorithm.

7.1 .Learning Classifier System with Genetic Algorithm.

In the run with GA, the genetic algorithm is invoked every 20 iterations and 80 percent of the current population undergoes reproduction, crossover, mutation, and replacement by crowding. Executing the KMB code, the system

responds by presenting the initial report display in **Appendix A.** For LCS-kick. The classifier system run for 100 iterations, termination with the snapshot report display in **Appendix A** for LCS-kick.

7.2 .Learning Classifier System without Genetic Algorithm.

The results without the genetic algorithm are presented in **Appendix B** for LCS-kick , here we see how the apportionment of credit algorithm adjusts the strength values of the rules, and in the long run the classifier system is able to sustain this relatively high level of performance even without genetic action. Results with the (GA) show improvement over the no-GA, both are illustrated in **Table. 5** and data chart illustrated in **Figure.6.**

Table.5: The KMB System Performance after 100 Iterations
Compare between classifier with genetic vs. classifier without genetic

Number of iterations	Proportion Correct using genetic algorithm	Proportion Correct without using Genetic algorithm
10	0.9000	0.8000
20	0.9567	0.8571
30	0.9681	0.8889
40	0.9775	0.9091
50	0.9844	0.9231
60	0.9900	0.9333
70	1.0000	0.9412
80	1.0000	0.9474
90	1.0000	0.9500
100	1.0000	1.000

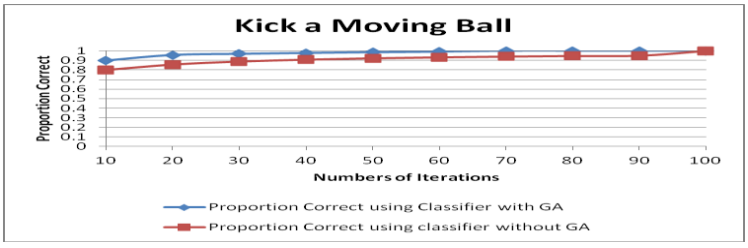

Figure.6: The KMB System Performance After100 Iterations
Compare Between Classifier with Genetic Vs. Classifier without Genetic

8. CONCLUSIONS

- Results show that a classifier system with genetic algorithm show improvement over the classifier system without genetic algorithm.
- The results without genetic algorithm show that the apportionment of credit algorithm adjust the strength values of rules.
- This paper shows how two different classifier systems are both able to demonstrate similar learning performance over a set of classification tasks.
- The time need to convergence is shorter when use CS with GA and it is more likely to achieve good result quickly.

9. REFERENCE

1. Holland, John H. (1986), "Escaping Brittleness: The possibilities of General - purpose Learning Algorithms Applied to Parallel Rule - based Systems", Machine Learning an Artificial Intelligence Approach Vol. II, pp. 593 - 623, ed. R.S. Michalski, J. G. Carbonnell and T. M. Mitchell, Tioga, Palo Alto, Calf.
2. Goldberg , David E. (1989), "Genetic Algorithms in Search, Optimization & Machine Learning"**,** Addison-WesleyPublishing Company, Inc.
3. Odetayo, Michael O. (1990), "On Genetic Algorithms in Machine Learning And Optimisation"**,** PhD Thesis,University of Strathclyde, Glasgow, U.K.
4. Amir Kharmandar, Alireza Naeimi, Alireza Molla Alizadeh, Shaghayegh Jafari, Samira Chavoshi,(2011)," Soccer Simulation 2DTeam Description Proposal for Robocup , ,Payame Noor University, Iran.

5. Brownlee Jason,(2007**) "**Learning Classifier Systems**",**Technical Report 070514A,Complex Intelligent Systems Laboratory, Centre for Information Technology Research, Faculty of Information and Communication Technologies, Swinburne University of Technology Melbourne, Australiajbrownlee@ict.swin.edu.au.
6. Ryan J. Urbanowicz and Jason H.Moore,(2009) "Learning Classifier Systems:A Complete Introduction, Review, and Roadmap"*Department of Genetics, Dartmouth College, Hanover, NH 03755, USA*Correspondence should be addressed to Jason H. Moore, jason.h.moore@dartmouth.edu.
7. Bull.Larry,(2004), "Learning Classifier Systems: A Brief Introduction", Faculty of Computing, Engineering & Mathematical Sciences University of the West of England, Bristol BS16 1QY, U.K. Larry.
8. Zhou. Qing Qing and Purvis. Martin,(2004) "A Market-Based Rule Learning System" aGuangDong Data Communication Bureau China Telecom 1 Dongyuanheng Rd., Yuexiunan, Guangzhou 510110, China, Department of Information Science, University of Otago, PO Box 56, Dunedin, New Zealand and/or improving the comprehensibility of the rules.
9. Jakobsen. Troels,(2004),"Classifier System.
10. Hartley Adrian R, (1999)" 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 Telephone Abstracts "Aarhus school of business ,Denmark. (+44) (0)121 414 3711.
11. Togelius. Julian,(2003) "Evolution of The Layers In a Subsumption Architecture Robot Controller" Dissertation for the Master of Science in Evolutionary and adaptive systems University of Sussex at Brighton.
12. Crook. Stamati.(2003) "Evolving expert systems for autonomous agent control using reinforcement learning." M.Sc Thesis, Evolutionary and Adaptive Systems. School of Cognitive and Computing Sciences. Sussex University.
13. Eriksson. Anders,(2002) "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.
14. Robert Elliott Smith , Max Kun Jiang ,Jaume Bacardit , Michael Stout , Natalio Krasnogor ,Jonathan D. Hirst,(2010)," A learning classifier system with mutual-information-based fitness", UK Engineering and Physical Sciences Research Council(EPSRC).

Appendix – A.A Kick Moving Ball Code FOR LCS – Kick Using classifier with GA

```
population parameters
number of classifiers      =      16
number of positions        =       3
bid coefficient            =  0.1000
bid spread                 =  0.0750
bidding tax                =  0.0100
existence tax              =  0.0200
generality probability     =  0.5000
bid specificity base       =  1.0000
bid specificity mult.      =  0.0000
edid specificity base      =  1.0000
ebid specificity mult.     =  0.0000
total number of bits   =       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    =       4
genetic algorithm parameters
proportion of select/gen    =  0.8000
Number of pairs to select   =       6
p mutation                  =  0.0200
p crossover                 =  1.0000
crowding factor             =       3
crowding population         =       3
snapshot repor
```

```
[block: iteration]  -  [0:0]
current Statius
signal              =    000
desired output      =000
classifier output   =000
environmental message:       000
no.       strength         bid       ebid    M   classifier
    1      10.00       0.00       0.00         000:[000]
    2      10.00       0.00       0.00         001:[001]
    3      10.00       0.00       0.00         010:[010]
    4      10.00       0.00       0.00         011:[011]
    5      10.00       0.00       0.00         100:[100]
    6      10.00       0.00       0.00         101:[101]
    7      10.00       0.00       0.00         110:[110]
    8      10.00       0.00       0.00         111:[111]
    9      10.00       0.00       0.00         ###:[000]
   10      10.00       0.00       0.00         ###:[001]
   11      10.00       0.00       0.00         ###:[010]
   12      10.00       0.00       0.00         ###:[011]
   13      10.00       0.00       0.00         ###:[100]
   14      10.00       0.00       0.00         ###:[101]
   15      10.00       0.00       0.00         ###:[110]
   16      10.00       0.00       0.00         ###:[111]
new winner[1] : old winner[1]
```

Initial report using classifier with genetic algorithm

```
snapshot report
[block: iteration]  -  [0:20]
current  Statius: signal=111,desired output=111,classifier output
=111,environmental message: 111
no.       strength      bid       ebid    M   classifier
    1      13.52       1.39       1.39    x    ###:[101]
    2      14.06       1.45       1.46    x    1#1:[100]
    3      10.38       0.00       0.00         000:[111]
    4      13.52       1.39       1.56    x    #11:[111]
    5      14.07       1.45       1.47    x    1#1:[111]
    6      14.07       1.45       1.47    x    111:[101]
    7      14.65       0.00       0.00         10#:[101]
    8      72.79       7.22       7.10    x    111:[111]
    9      14.06       0.00       0.00         101:[100]
   10       9.96       1.03       1.08    x    #11:[010]
   11      13.52       1.39       1.33    x    #11:[111]
   12      13.50       1.39       1.33    x    111:[111]
   13      14.63       1.51       1.54    x    1##:[101]
   14      14.04       0.00       0.00         101:[100]
   15      14.63       1.51       1.50    x    #11:[111]
   16      10.35       1.07       0.98    x    11#:[010]
new winner[8] : old winner[8]
genetic algorithm report
 pair mate1 mate2  sitecross  mort1 mort2
  1      4       6       2     16      16
  2     15       6       2     12      11
  3     13      14       2      9      10
  4     14      15       3     16       3
  5      5       8       3      2       6
  6      8       6       1      1      14
```

Last report using classifier with genetic algorithm

Appendix – B.A Kick Moving Ball Code FOR LCS – Kick Using classifier without GA

```
snapshot report
signal =  000,desired output =000, classifier output  =000
environmental message:       000
```

no.	strength	bid	ebid	M	classifier
1	10.00	0.00	0.00		000:[000]
2	10.00	0.00	0.00		001:[001]
3	10.00	0.00	0.00		010:[010]
4	10.00	0.00	0.00		011:[011]
5	10.00	0.00	0.00		100:[100]
6	10.00	0.00	0.00		101:[101]
7	10.00	0.00	0.00		110:[110]
8	10.00	0.00	0.00		111:[111]
9	10.00	0.00	0.00		###:[000]
10	10.00	0.00	0.00		###:[001]
11	10.00	0.00	0.00		###:[010]
12	10.00	0.00	0.00		###:[011]
13	10.00	0.00	0.00		###:[100]
14	10.00	0.00	0.00		###:[101]
15	10.00	0.00	0.00		###:[110]
16	10.00	0.00	0.00		###:[111]

new winner[1] : old winner[1]

Initial report using classifier without genetic algorithm

snapshot report

signal= 111 ,desired output =111, classifier output=111, environmental message:111

no.	strength	bid	ebid	M	classifier
1	8.68	0.00	0.00		000:[000]
2	8.68	0.00	0.00		001:[001]
3	8.68	0.00	0.00		010:[010]
4	8.51	0.00	0.00		01#:[001]
5	8.68	0.00	0.00		100:[100]
6	8.68	0.00	0.00		101:[101]
7	8.68	0.00	0.00		110:[110]
8	51.68	4.79	4.82	x	111:[111]
9	8.59	0.00	0.00		011:[011]
10	8.51	0.00	0.00		##0:[010]
11	8.25	0.85	0.80	x	###:[101]
12	8.68	0.00	0.00		011:[011]
13	21.73	2.24	2.12	x	1##:[101]
14	8.51	0.00	0.00		101:[100]
15	21.73	2.24	2.26	x	#11:[111]
16	8.68	0.00	0.00		011:[011]

new winner[8] : old winner[8]

Last report using classifier without genetic algorithm