Comparing Perfect and Default Hierarchal Rules in solving Animate Problem

Lubna Zaghlul Bashir
Department of Building and Construction Engineering
University of Technology

Abstract:

The animate problem is the problem faced by an artificial animal that has to learn how to survive in its environment. Behaviors are specified through differential equations, forming a global system made of behavior subsystems, which interact in a number of ways. In this work, subsystems interact together by using learning classifier systems. , to Illustrates the approach with the example of Change Chameleon Color (CCC). The *CCC* System is built of two-classifier subsystems working together, each classifier system teaches a simple behavior, first classifier system, simulated robot chase behavior i.e. teach robot to move single step toward moving Light, second classifier system, teaches the simulated robot mimetic behavior i.e. teaches robot to change its color according to the background color. The system as a whole has as its learning goal the coordinate of behaviors. The results show identifying and coding appropriate activation schemes are decisive for the performance of a control system and Learning Classifier Systems is a feasible tool to build robust simulated robot control system. A Simple experiment was executed for *CCC* Compare between using perfect and default hierarchical rules, and the results show that the set of rules working together, as a default hierarchy should perform as well as the perfect set of rules. Default hierarchies' rules, containing fewer rules, than non-overlapping rule sets for the same problem. Default hierarchical rules also enlarge the set of solutions space with no increase in the size of the problem space.

Keywords: Animate, Robotics behavior, Default Hierarchies, Learning Classifier Systems.

مقارنة بين القوانين التامة والهرمية لحل مشكلة البقاء

م. لبنى زغلول بشير
الجامعة التكنولوجية -قسم هندسة البناء و الإنشاءات

المستخلص :
إن مشكلة سلوكيات البقاء هي مشكلة واجهت الكائنات الذكية التي يجب عليها التعلم من اجل البقاء في البيئة المحيطة.السلوك يحدد من خلال معادلات مختلفة، السلوكيات الجزئية تتفاعل مع بعضها باستخدام عدة طرق. في هذا العمل استخدمنا أنظمة التصنيف التعليمية لتشكل الحركة الكلية للنظام.هذا العمل يوضح الأسلوب من خلال النظام (CCC) Change Chameleon Color وهو كائن آلي مزيف يقوم بمتابعة مصدر الضوء و يقوم بتغيير لونه نسبة الى لون الخلفية للبيئة .النظام (*CCC*) صمم باستخدام اثنين من أنظمة التصنيف التعليمية ، نظام التصنيف الأول يستخدم لتعليم الكائن الآلي المزيف سلوك الاصطياد وهو أن يتحرك خطوة واحدة باتجاه مصدر الضوء المتحرك ونظام التصنيف الثاني يعلم الكائن الآلي سلوك التقليد وهو أن يغير لونه حسب خلفية لون البيئة وهدف هذا النظام هو التنسيق بين الفعاليات.النتائج بينت أن تعريف و برمجة فعالية حركية تكون حاسمة لكفاءة نظام سيطرة وأن أنظمة التصنيف التعليمية أداة ملائمة لبناء نظام سيطرة لكائن آلي مزيف.
تم إجراء مقارنة بين استخدام القوانين التامة والقوانين الهرمية ، النتائج بينت ان مجموعة القوانين الهرمية تعمل بكفاءة مقاربة لكفاءة القوانين التامة و تضم مجموعة اقل من القوانين كما انها توسع من فضاء الحل دون ان يزداد حجم المشكلة.

1. Animate problem

The animate problem is the problem faced by an artificial animal that has to learn how to survive in its environment. Wilson propose the following reasons to explain why this is a difficult problem: information is difficult to classify because there is no a priori knowledge about how to relate environmental situations ,i.e. information coming from the environment, to actions that can take the animate closer to the goal state and also because there is no teacher that can correct useless or wrong actions. another motive of difficulty for the learning animate is the stage setting problem: how can the animate realize that a particular action, although not directly rewarded, is important because it is a necessary step on the rout to the goal? Wilson says that the animate problem can be summarized as "the problem of incrementally learning multiple disjunctive concepts under payoff" facing this problem by provide the feasibility of classifier systems as a tool to learn disjunctive concepts under payoff.[1].

2. Behavior

It becomes natural to think of human beings as information processing systems that receive input from the environment (perception), process that information (thinking), and act upon the decision reached (behavior), thus *Behavior* is a product of the interaction between an agent and its environment. Five kinds of basic behaviors considers :

1. The *approaching behavior*, i.e. getting closer to an almost *still* object with given features; in the natural world, this response is a fundamental component of feeding and sexual behavior.
2. The *chasing behavior*, i.e. following and trying to catch a *moving* object with given features; as the preceding approaching behavior, this response is important for feeding and reproduction.
3. The *mimetic behavior*, i.e. entering a well-defined physical state which is a function of a feature of the environment; this is inspired by the natural behavior of a chameleon, changing its color according to the color of the environment.
4. The *avoidance behavior*, i.e. avoiding physical contact with an object of a given kind, this can be seen as the artificial counterpart of a behavioral pattern which allows an organisms to avoid hurting objects.
5. The *escaping behavior*, i.e. moving as far as possible from an object with given features; the object can be viewed as a predator.[2].

3. 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).[3,4,5]

3.1 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:

$$C_1, C_2, ..., C_n/a$$

Where $C_1,C_2,\ldots C_n$ {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}.[3,6].

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. [6,7,8].

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. [7].

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. [7]

3.2 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:

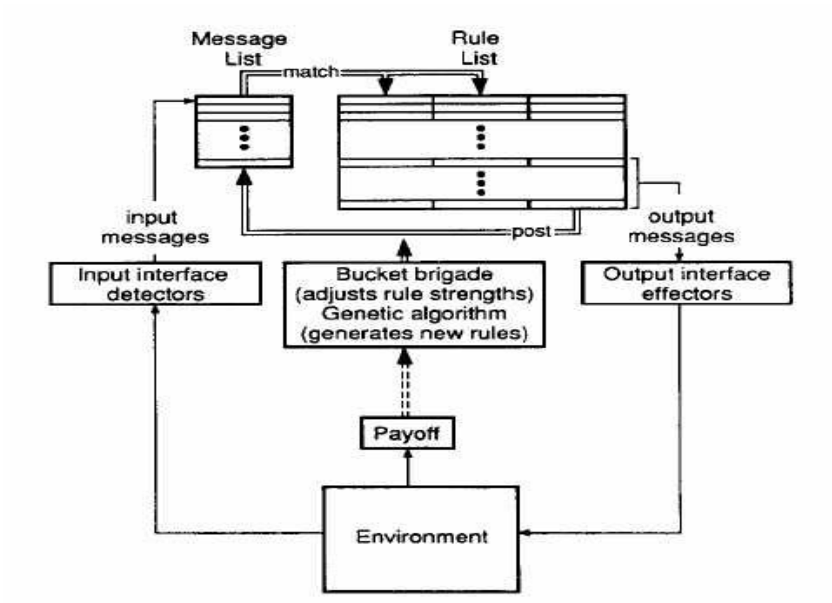
$$\text{Bid} = C_{bid}*\text{strength}*\text{specificity}.$$

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. [9,10].

3.3 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 Fig.1. [4,11,12].**


Message List
Rule List
match
post
input messages
output messages
Input interface detectors
Bucket brigade (adjusts rule strengths) Genetic algorithm (generates new rules)
Output interface effectors
Payoff
Environment

4. Fig.1.Simplified architecture of learning classifier system

4.default hierarchies rules (dh)

Learning classifier systems have the feature that they can autonomously organize rules into hierarchies. A default hierarchy is a multi – level structure in which classifiers become more general as the top level is ascended. Each general rule responds to a broad set of environmental messages, so that just a few rules can cover all possible states of the environment. Since a general rule may respond in the same way to many inputs that do not really belong in the same category, it will often make error. To correct the mistakes made by the general classifiers, lower level, exception rules evolve in the default hierarchy. The lower level classifiers are more specific than the higher level rules, each exception rule responds to a subset of the situations covered by the more general rule, but it also makes fewer mistakes than the default rules made on those states. For Example consider the following classifiers of length L=3:

1##: action1
10#: action2
101: action3

These classifiers define a simple three level default hierarchy, in which the first classifier is the most general, covering four messages,

the second is an exception to the first, covering two of those four messages, and the third is an exception to both, covering just one message.

Hierarchical rule set structure enables[9]:

1. Improved parsimony in rule sets, in that they use fewer rules than homomorphic structures.
2. Expansion of the solution space, additional exception rules can be added to obtain the required degree of accuracy.
3. Graceful refinement of rule sets through the structured addition of exception rules.

5. Change chameleon color: a case study

In this study, the system called Change Chameleon Color *(CCC)* uses two classifier systems, which has flat architecture. In the system the robot should learn complex behavior consisting of two basic behaviors: Chase behavior and mimetic 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 light (chase behavior).second classifier called (LCS-Color) learn the robot change its color according to environment background color (mimetic behavior). The objects in environment are as follows: Moving Light , moving Robot, and the environment was partitioned into eight different colors The environment illustrated in Fig.2.

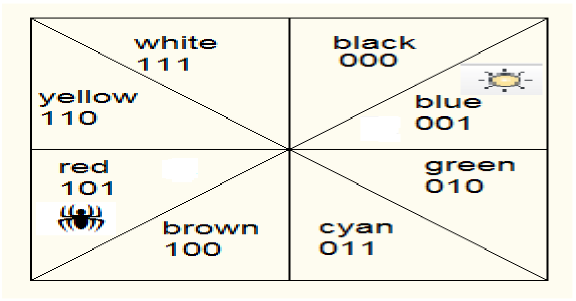


Fig.2: The Change Chameleon Color Environment.

5.1 (CCC) System Structure

The CCC system is built of two learning classifier systems. Organized in flat architecture, interacting to gather to perform complex behavior, consist of two classifier systems (LCS-Position) and (LCS-Color).the CCC structure is shown in Fig.3.

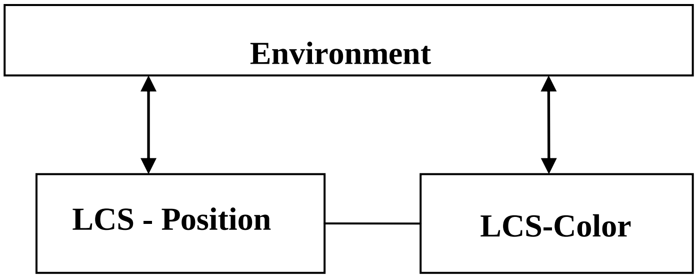


Fig.3: CCC System Structure.

5.2.(LCS-Position) development

LCS-Position is used to learn robot move single step toward moving Light. The movement capability is completely symmetric a long the two axis .the direction of movement is illustrated in Fig.4.

Northwest (111)	**North (000)**	**Northeast (001)**
West (110)	🕷	**East (010)**
Southwest (101)	**South (100)**	**Southeast (011)**

Fig.4: Direction of robot movement

5.2.1 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 Light 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 light from robot is to north
0 0 1	Relative position of light from robot is to north - east
0 1 0	Relative position of light from robot is to east
0 1 1	Relative position of light from robot is to south - east
1 0 0	Relative position of light from robot is to south
1 0 1	Relative position of light from robot is to south - west
1 1 0	Relative position of light from robot is to west
1 1 1	Relative position of light from robot is to north - west

5.2.2 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

5.2.3 Representation of (LCS – Position)

LCS – Position consists of a condition part of (3bit) representing the position of Light 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 Light 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 Light From the robot / direction moving of robot
0 0 0 / 0 0 0

5.3 (LCS – Color) development

LCS – Color is used to learn simulated robot change its color according to environment background color in mimetic behavior.

5.3.1 Coding (LCS – Color) Conditions

The length of message, which LCS – Color is received is always, 3 bit environment message mapping it to eight sates from 0 to 7 of three bits only. The meaning of three bit in input message of LCS – Color determines the environment background color. The form and meaning of three bit LCS – Color is shown in Table 3.

Table 3: form and meaning of LCS - Color Conditions

The message	Its meaning
0 0 0	Environment color is black
0 0 1	Environment color is blue
0 1 0	Environment color is green
0 1 1	Environment color is cyan
1 0 0	Environment color is brown
1 0 1	Environment color is red
1 1 0	Environment color is yellow
1 1 1	Environment color is white

5.3.2 Coding (LCS – Color) 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 – Color actions

The action	Its meaning
0 0 0	Means robot change its color to black
0 0 1	Means robot change its color to blue
0 1 0	Means robot change its color to green
0 1 1	Means robot change its color to cyan
1 0 0	Means robot change its color to brown
1 0 1	Means robot change its color to red
1 1 0	Means robot change its color to yellow
1 1 1	Means robot change its color to white

5.3.3 Representation of (LCS – Color)

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

Example:

The representation of the rule "if a environment background color is blue then the action to be taken by robot is to change its color to blue color, and so on.

Environment background color is blue / Robot change its color to blue

0 0 1 / 0 0 1

6. The performance system for (ccc) 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 CCC 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.

7. Apportionment of credit (aoc) for (ccc) system

The procedure (AOC), in the system (CCC) 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. 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 CCC system, a classifier's bid is defined as a product of C_{bid} and a linear function of classifier's specificity, and classifier strength [6].

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

$$f(SP) = bid_1 + bid_2 * SP \tag{2}$$

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:

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

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:

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

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 CCC 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:**

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

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:

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

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 [13,14]:

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

The apportionment of credit equation recasts into a more useful form where all payments and taxes have been replaced by their strength equivalent [6].

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

8. Rule discovery for (ccc) system

GA is a way of injecting new possibly better rules into the system (CCC) 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 (CCC) 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 *Fig. 5.***
- **The selection process for (CCC) 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 (CCC) 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.**

```
Procedure GA;
{Coordinate selection, mating, crossover, mutation, & replacement}
Begin with population do
     begin
       Statistics (population);      {get average, max,min, sumstrength}
       For j:=1 to nselect 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;
```

Fig. 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) [6,13].

9.executing of system change chameleon color code

The whole project was programmed in Pascal language, Executing the change chameleon color code, the system responds by presenting the initial report display in Appendix A for (LCS – Position) .the classifier system run for 50 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. Executing the change chameleon color code, the system responds by presenting the initial report display in Appendix B for (LCS – Color) .the classifier system run for 50 iterations, termination with the last snapshot report display in Appendix B 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.
In the change chameleon color (CCC) code simple experiment was implements in simulation world: compare between using perfect rules and default hierarchal rules.

9.1 The Perfect Rules

In CCC code an experiment were performs using the perfect set of rules:
The perfect rules in (LCS- color and LCS-position) consist of 8 rules, each rule has eight actions. Executing the CCC code, the system responds by presenting the initial report display in Appendix C. for LCS-Position. The classifier system run for 50 iterations, termination with the snapshot report display in Appendix C for LCS-Position. The correct rules have achieved high strength values, by contrast the bad rules have strength and bid values near zero. The classifier system eliminates the bad rule quickly thereby achieving near perfect performance.

9.2 The Default Hierarchy (DH) Rules

In the situation when have less than perfect rules in the classifier population, the classifiers organize themselves into a default hierarchy structure .In a default hierarchy general rule (those with many #'s) cover the general conditions and more specific, possibly overlapping rules cover the exception.

In CCC system problem, consider the set of default hierarchy rule shown in Appendix D. for LCS-position.

Assume that each rule receives equal pay off when rewarded, and if further assume that when two overlapping rules bid to answer a particular message, the more specific rules wins, this set of rule constitutes a working a default hierarchy.

In fact the set of rules working together, as a default hierarchy should perform as well as the perfect set of rules. The last report is illustrated in Appendix D. For LCS-position.

The performance of the system after 50 iteration compare between perfect rules vs. Default Hierarchal rules is illustrated in Table. 5 and data chart illustrated in Fig.6.

Table.5 : The CCC System Performance After 50 Iterations Compare between Perfect rules vs. Default Hierarchal rules

Number of iteration	Proportion Correct using Perfect Rules	Proportion Correct using default Hierarchal Rules
5	0.9000	0.8000
10	0.9167	0.8571
15	0.9281	0.8889
20	0.9375	0.9091
25	0.9444	0.9231
30	0.9500	0.9333
35	1.0000	0.9412
40	1.0000	0.9474
45	1.0000	0.9500
50	1.0000	1.000

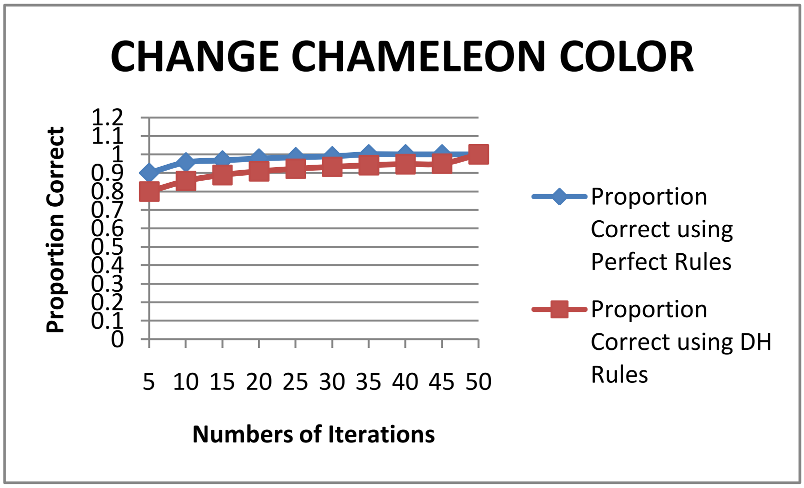


Fig.6 : the CCC system performance after50 iterations Compare between Perfect rules vs. Default Hierarchal rules

10. Conclusions:

- **Learning Classifier Systems is a feasible tool to build robust simulated robot control system.**
- **Experimentally shows that the set of rules working together, as a default hierarchy should perform as well as the perfect set of rules. Default hierarchies' rules, containing fewer rules, than non-overlapping rule sets for the same problem. Default hierarchical also enlarges the set of solutions space with no increase in the size of the problem space.**
- **The system was able to learn rules for the given task using only a few training examples and starting with classifiers that were randomly generated.**
- **This paper shows how two different classifier systems are both able to demonstrate similar learning performance over a set of classification tasks.**

APEENDIX _ A

```
========================================
 Change Chameleon Color Code for LCS - Position
========================================
population parameters
--------------------
number of classifiers     =        50
number of positions       =         3
number of action          =         3
bid coefficient           =  0.1000
bid spread                =  0.0750
bidding tax               =  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 =  0.4000
Number to select          =        10
Mutation probability      =  0.0200
Crossover probability     =  0.8000
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##:[000]
   2     10.00      0.00     0.00       0##:[001]
   3     10.00      0.00     0.00       0##:[010]
   4     10.00      0.00     0.00       0##:[011]
   5     10.00      0.00     0.00       1##:[100]
   6     10.00      0.00     0.00       1##:[101]
   7     10.00      0.00     0.00       1##:[110]
   8     10.00      0.00     0.00       1##:[111]
   9     10.00      0.00     0.00       #0#:[000]
  10     10.00      0.00     0.00       #0#:[001]
  11     10.00      0.00     0.00       #0#:[100]
  12     10.00      0.00     0.00       #0#:[101]
  13     10.00      0.00     0.00       #1#:[010]
  14     10.00      0.00     0.00       #1#:[011]
  15     10.00      0.00     0.00       #1#:[110]
  16    10.00     0.00      0.00       #1#:[111]
  17    10.00     0.00      0.00       ##0:[000]
  18    10.00     0.00      0.00       ##0:[010]
  19    10.00     0.00      0.00       ##0:[100]
  20    10.00     0.00      0.00       ##0:[110]
  21    10.00     0.00      0.00       ##1:[001]
  22    10.00     0.00      0.00       ##1:[011]
  23    10.00     0.00      0.00       ##1:[101]
  24    10.00     0.00      0.00       ##1:[111]
  25    10.00     0.00      0.00       ###:[000]
  26    10.00     0.00      0.00       ###:[001]
  27    10.00     0.00      0.00       ###:[010]
  28    10.00     0.00      0.00       ###:[011]
  29    10.00     0.00      0.00       ###:[100]
  30    10.00     0.00      0.00       ###:[101]
  31    10.00     0.00      0.00       ###:[110]
  32    10.00     0.00      0.00       ###:[111]
  33    10.00     0.00      0.00       00#:[001]
  34    10.00     0.00      0.00       00#:[000]
  35    10.00     0.00      0.00       #00:[000]
  36    10.00     0.00      0.00       #00:[100]
  37    10.00     0.00      0.00       0#0:[000]
  38    10.00     0.00      0.00       0#0:[010]
  39    10.00     0.00      0.00       11#:[110]
  40    10.00     0.00      0.00       11#:[111]
  41    10.00     0.00      0.00       #11:[011]
  42    10.00     0.00      0.00       #11:[111]
  43    10.00     0.00      0.00       1#1:[101]
  44    10.00     0.00      0.00       1#1:[111]
  45    10.00     0.00      0.00       01#:[010]
  46    10.00     0.00      0.00       01#:[011]
  47    10.00     0.00      0.00       #01:[001]
  48    10.00     0.00      0.00       #01:[101]
  49    10.00     0.00      0.00       0#1:[001]
  50    10.00     0.00      0.00       0#1:[001]
new winner[1] : old winner[1]
```

Initial report for CCC System for LCS – Position

```
Snapshot report
---------------
[block: iteration]  -  [0:26]
current  status
----------------
signal     =    010
Decoded signal    =      2
desired output    =      2
```

```
classifier output  =      2
environmental message:      010
no.  strength     bid      ebid  M     classifier
-----------------------------------------------------
   1     59.51     0.00     0.00          00#:[010]
   2     55.45     1.43     1.52   x     ###:[010]
   3     59.63     2.31     2.21   x      0##:[010]
   4     57.42     0.00     0.00          011:[010]
   5     59.04     2.28     2.30   x      0##:[010]
   6     62.57     2.42     2.45   x      0##:[001]
   7     63.96     2.47     2.43   x      0##:[011]
   8     58.67     3.78     3.91   x      010:[010]
   9     64.98     2.51     2.47   x      0##:[010]
  10     52.69     0.00     0.00          1#1:[101]
  11     66.28     2.56     2.59   x      0##:[010]
  12     62.23     0.00     0.00         00#:[010]
  13     56.38     0.00     0.00         001:[101]
  14     72.35     3.73     3.83   x     01#:[010]
  15     57.66     2.97     2.97   x     01#:[010]
  16     44.01     0.00     0.00         011:[101]
  17     60.07     0.00     0.00         001:[011]
  18     60.73     3.13     3.19   x     01#:[010]
  19     59.79     2.31     2.31   x     0##:[010]
  20     70.86     0.00     0.00         00#:[010]
  21     76.86     2.97     3.01   x     0##:[010]
  22     59.47     0.00     0.00         0#1:[001]
  23     98.79     5.09     5.00   x     01#:[010]
  24     60.26     2.33     2.46   x     0##:[010]
  25     62.67     2.42     2.34   x     0##:[011]
  26     81.38     3.15     3.10   x     0##:[011]
  27     67.77     3.49     3.41   x     0#0:[010]
  28     58.34     3.01     3.04   x     01#:[010]
  29     72.35     2.80     2.76   x     0##:[010]
  30     81.69     4.94     5.02   x     010:[010]
  31     62.67     3.23     3.35   x     01#:[010]
  32     66.28     2.56     2.55   x     0##:[010]
  33     64.98     1.67     1.84   x     ###:[010]
  34     44.01     0.00     0.00         011:[101]
  35     81.38     4.19     4.26   x     01#:[010]
  36     67.98     2.63     2.67   x     0##:[010]
  37     76.86     3.96     3.99   x     01#:[010]
  38     58.05     0.00     0.00         001:[010]
  39     57.18     0.00     0.00         0#1:[010]
  40     69.42     2.68     2.58   x     0##:[011]
  41     60.22     2.33     2.29   x     0##:[010]
  42     57.45     1.48     1.55   x     ###:[001]
  43     57.92     0.00     0.00         011:[011]
  44    76.63     2.96     2.94   x    #1#:[000]
  45    56.38     0.00     0.00        1#1:[011]
  46    63.26     0.00     0.00        00#:[011]
  47    54.46     3.51     3.47   x    010:[000]
  48    64.86     2.51     2.48   x    0##:[010]
  49    67.98     3.50     3.42   x    01#:[010]
  50    59.51     0.00     0.00        011:[011]
new winner[30] : old winner[23]
```

Last report for CCC System for LCS - Position

APPENDIX _ B

```
========================================
 Change Chameleon Color Code FOR LCS - Color
========================================
population parameters
---------------------
number of classifiers      =      50
number of positions        =       3
number of action           =       3
bid coefficient            =  0.1000
bid spread                 =  0.0750
bidding tax                =  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              =     15
Mutation probability     =  0.0200
Crossover probability   =  0.8000
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##:[000]
    2      10.00      0.00      0.00       0##:[001]
    3      10.00      0.00      0.00       0##:[010]
    4      10.00      0.00      0.00       0##:[011]
    5      10.00      0.00      0.00       1##:[100]
    6      10.00      0.00      0.00       1##:[101]
    7      10.00      0.00      0.00       1##:[110]
    8      10.00      0.00      0.00       1##:[111]
    9      10.00      0.00      0.00       #0#:[000]
   10      10.00      0.00      0.00       #0#:[001]
   11      10.00      0.00      0.00       #0#:[100]
   12      10.00      0.00      0.00       #0#:[101]
   13      10.00      0.00      0.00       #1#:[010]
   14      10.00      0.00      0.00       #1#:[011]
   15      10.00      0.00      0.00       #1#:[110]
   16      10.00      0.00      0.00       #1#:[111]
   17      10.00      0.00      0.00       ##0:[000]
   18      10.00      0.00      0.00       ##0:[010]
   19      10.00      0.00      0.00       ##0:[100]
   20      10.00      0.00      0.00       ##0:[110]
   21      10.00      0.00      0.00       ##1:[001]
   22      10.00      0.00      0.00       ##1:[011]
   23      10.00      0.00      0.00       ##1:[101]
   24      10.00      0.00      0.00       ##1:[111]
   25      10.00      0.00      0.00       ###:[000]
```

```
  26    10.00     0.00     0.00      ###:[001]
  27    10.00     0.00     0.00      ###:[010]
  28    10.00     0.00     0.00      ###:[011]
  29    10.00     0.00     0.00      ###:[100]
  30    10.00     0.00     0.00      ###:[101]
  31    10.00     0.00     0.00      ###:[110]
  32    10.00     0.00     0.00      ###:[111]
  33    10.00     0.00     0.00      00#:[001]
  34    10.00     0.00     0.00      00#:[000]
  35    10.00     0.00     0.00      #00:[000]
  36    10.00     0.00     0.00      #00:[100]
  37    10.00     0.00     0.00      0#0:[000]
  38    10.00     0.00     0.00      0#0:[010]
  39    10.00     0.00     0.00      11#:[110]
  40    10.00     0.00     0.00      11#:[111]
  41    10.00     0.00     0.00      #11:[011]
  42    10.00     0.00     0.00      #11:[111]
  43    10.00     0.00     0.00      1#1:[101]
  44    10.00     0.00     0.00      1#1:[111]
  45    10.00     0.00     0.00      01#:[010]
  46    10.00     0.00     0.00      01#:[011]
  47    10.00     0.00     0.00      #01:[001]
  48    10.00     0.00     0.00      #01:[101]
  49    10.00     0.00     0.00      0#1:[001]
  50    10.00     0.00     0.00      0#1:[001]
new winner[1] : old winner[1]

Initial report for CCC System for LCS –Color

snapshot report
---------------
[block: iteration]  -  [0:22]
current  status
----------------
signal     =    101
Decoded signal    =      5
desired output     =      5
classifier output  =      5
environmental message:      101
no.     strength       bid      ebid   M   classifier
-------------------------------------------------------
   1     75.93     4.89     4.88   x    101:[101]
   2     67.98     0.00     0.00        #00:[101]
   3     54.69     2.11     2.16   x    ##1:[101]
   4     65.10     0.00     0.00        0#1:[000]
   5     71.79     3.70     3.85   x    #01:[101]
   6     74.23     0.00     0.00        #00:[101]
   7   64.43    3.32    3.22  x   #01:[101]
   8   71.79    3.70    3.76  x   #01:[101]
   9   55.08    0.00    0.00      00#:[001]
  10   66.58    0.00    0.00      #00:[101]
  11   87.83    5.36    5.33  x   101:[101]
  12   64.07    3.30    3.22  x   #01:[000]
  13   57.25    2.95    2.91  x   #01:[000]
  14   70.22    3.62    3.63  x   #01:[001]
  15   59.38    3.06    3.24  x   #01:[000]
  16   64.49    3.32    3.22  x   #01:[101]
  17   62.77    0.00    0.00      #10:[001]
  18   62.61    3.23    3.25  x   1#1:[101]
  19   65.34    3.37    3.43  x   #01:[101]
  20   68.68    3.54    3.51  x   #01:[101]
  21   70.22    3.62    3.54  x   #01:[101]
  22   73.08    3.77    3.78  x   1#1:[101]
  23   63.42    0.00    0.00      #11:[000]
  24   71.92    0.00    0.00      #00:[101]
  25   65.47    0.00    0.00      #11:[000]
  26   58.16    3.00    2.98  x   #01:[101]
  27   66.32    0.00    0.00      #00:[101]
  28   74.91    3.86    3.87  x   #01:[101]
  29   70.52    3.63    3.66  x   #01:[101]
  30   66.01    0.00    0.00      #00:[101]
  31   70.52    3.63    3.59  x   #01:[000]
  32   68.68    3.54    3.48  x   #01:[101]
  33   71.92    0.00    0.00      #11:[101]
  34   66.11    0.00    0.00      #00:[001]
  35   74.91    2.90    2.88  x   ##1:[101]
  36   65.87    0.00    0.00      001:[001]
  37   65.47    0.00    0.00      0#1:[101]
  38   69.63    0.00    0.00      001:[101]
  39   83.19    4.29    4.31  x   #01:[101]
  40   67.29    3.47    3.53  x   #01:[101]
  41   41.97    0.00    0.00      00#:[110]
  42   70.52    3.63    3.65  x   #01:[101]
  43   63.64    0.00    0.00      #00:[101]
  44   56.84    2.20    2.21  x   #0#:[110]
  45   73.08    3.77    3.83  x   #01:[001]
  46   64.77    3.34    3.40  x   #01:[101]
  47   75.93    3.91    3.93  x   #01:[101]
  48  101.89    5.25    5.26  x   #01:[101]
  49   65.83    0.00    0.00      #11:[000]
  50   65.68    0.00    0.00      #11:[000]
new winner[11] : old winner[48]
```

Last report for CCC System for LCS – Color

Appendix – C

```
=========================================================
Change Chameleon Color Code FOR LCS – Position Using Perfect Rules
=========================================================
snapshot report
[block: iteration]  -  [0:0]
current  Statius
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      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]
```

Initial report using perfect rules

```
snapshot report
[block: iteration]  -  [0:10]
current  Statius
signal          =   111
desired output    =111
classifier output  =111
environmental message:     111
no.    strength      bid     ebid   M   classifier
-------------------------------------------------------
    1     9.80          0.00     0.00      000:[000]
    2     9.80          0.00     0.00      001:[001]
    3     9.80          0.00     0.00      010:[010]
    4     9.80          0.00     0.00      011:[011]
    5     9.80          0.00     0.00      100:[100]
    6     9.80          0.00     0.00      101:[101]
    7     9.80          0.00     0.00      110:[110]
    8    19.07          0.63     0.66   x  111:[111]
new winner[8] : old winner[8]
```

Last report using perfect rules

Appendix – D

```
=============================================================
Change Chameleon Color Code FOR LCS – Position Using Default Hierarchal
Rules
=============================================================
snapshot report
[block: iteration]  -  [0:0]
current  Statius
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##:[000]
    2    10.00     0.00     0.00       0##:[001]
    3    10.00     0.00     0.00       0##:[010]
    4    10.00     0.00     0.00       101:[101]
    5    10.00     0.00     0.00       0##:[011]
    6    10.00     0.00     0.00       010:[010]
    7    10.00     0.00     0.00       1##:[100]
    8    10.00     0.00     0.00       1##:[110]
    9    10.00     0.00     0.00       1##:[111]
   10    10.00     0.00     0.00       1##:[101]
   11    10.00     0.00     0.00       #0#:[000]
   12    10.00     0.00     0.00       110:[110]
   13    10.00     0.00     0.00       #0#:[001]
   14    10.00     0.00     0.00       001:[001]
   15    10.00     0.00     0.00       #0#:[100]
   16    10.00     0.00     0.00       111:[111]
```

Initial report using DH rules

```
snapshot report
[block: iteration]  -  [0:11]
current  Statius
signal              =   111
desired output    =111
classifier output  =111
environmental message:     111
no.     strength      bid     ebid   M   classifier
---------------------------------------------------
    1     9.80     0.00     0.00        0##:[000]
    2     9.80     0.00     0.00        0##:[001]
    3     9.80     0.00     0.00        0##:[010]
    4     9.80     0.00     0.00        101:[101]
    5     9.80     0.00     0.00        0##:[011]
```

```
   6     9.80     0.00     0.00          010:[010]
   7     9.70     0.38     0.34    x    1##:[100]
   8     9.70     0.38     0.37    x    1##:[110]
   9     9.70     0.38     0.33    x    1##:[111]
  10      9.70     0.38     0.38   x    1##:[101]
  11      9.80     0.00     0.00        #0#:[000]
  12      9.80     0.00     0.00         110:[110]
  13      9.80     0.00     0.00        #0#:[001]
  14      9.80     0.00     0.00        001:[001]
  15      9.80     0.00     0.00       #0#:[100]
  16    19.07      0.63      0.64   x  111:[111]
new winner[16] : old winner[16]
```

Last report using DH rules

11.References:

1. [Goldberg, David E.1989] "Genetic Algorithm in Search, Optimization, and Machine Learning", Addison Wesley Longmont, International Student Edition.
2. [Dorigo. Marco. and Colombetti. Marco,1994] "Training Agents to Perform Sequential Behavior" International Computer Science Institute, to appear in adaptive behavior, MIT press.
3. [Amir Kharmandar, Alireza Naeimi, Alireza Molla Alizadeh, Shaghayegh Jafari, Samira Chavoshi,2011]," Soccer Simulation 2DTeam Description Proposal for Robocup , ,Payame Noor University, Iran.
4. [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.
5. [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.
6. [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.

7. [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.
8. [Jakobsen. Troels,2004],"Classifier System
9. [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.
10. [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.
11. [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.
12. [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.
13. [Odetayo. Michael O,1990]."Machine learning using a genetic – based approach." School of Mathematical and Information Sciences.
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).