Eng. & Technology, Vol.24, No.6, 2005

# Robot Behavior Using Distributed Learning Classifier System

Dr. Aladdin Jamel Abdulwahid Al_Naji*

Received on : 24/12/2002
Accepted on :28/6/2005

Abstract
Robot has become such a prominent tool that it has increasingly taken a more important role in many different industries because it operates with great efficiency and accuracy.
In this paper genetic algorithms are used to learn complex behaviors for robot using distributed learning classifier systems as a control system.
To investigate this goal is proposed a simulation system. This system uses a simulated environment with simulated robots. Because it takes far less time to run experiments in simulated world than in the real world, a much greater number of design possibilities can be tested. Also simulated environment have proved very useful to test design options, and the results of simulations turned out to be robust enough to be carried over to the real world without major problems.

**Keywords:** Robotics, classifier system, machine learning.

سلوك الانسان الالي باستخدام نظام التصنيف التعلمي الموزع

الخلاصة
الروبوت اصبح من الادوات البارزة التي لها دور مهم اخذ بالازدياد اكثر فاكثر في العديد من الصناعات لانه يعمل بكفاءة كبيرة ودقة عالية.
في هذا البحث تم استخدام الخوارزمية الجينية لتعليم الروبوت سلوكيات معقدة باستخدام انظمة التصنيف الموزعة كنظام سيطرة.
لتحقيق هذا الهدف تم افتراض نظام محاكاة. في هذا النظام تم استخدام محيط محاكي مع روبوتات محاكية. لانها تاخذ فترة زمنية قليلة لعمل التجارب في العالم المحاكي عما هو عليه في العالم الحقيقي كما ان عدد كبير جدا من التصاميم المحتملة يمكن ان تختبر وايضا ان المحيط المحاكى مفيد جدا لاختبار خيارات التصميم، ونتائج المحاكاة يمكن تثبيتها في النهاية لان تكون كافية القوة لتنقل الى العالم الحقيقي بدون مشاكل كبيرة.

## 1.Introduction

Genetic algorithms are adaptive search techniques that can learn high performance knowledge structures. The genetic algorithms' strength come from the implicitly parallel search of the solution space that it performs via a population of candidate solutions and this population is manipulated in the simulation. The candidate solutions represent every possible behavior of the robot and based on the overall performance of the candidates, each could be assigned a fitness value. Genetic operators could then be applied to improve the performance of the population of the behaviors. One cycle of testing all the competing behavior is defined as a generation, and is repeated until a good behaviors is evolved. The good behavior is then applied to the real world. Also

* Dept. of Computer Science, University of Technology, Baghdad, IRAQ.

https://doi.org/10.30684/etj.24.6.1
University of Technology-Iraq, Baghdad, Iraq/2412-0758

because of the nature of genetic algorithm, the initial knowledge does not have to be very good.

The robot are used to perform increasingly difficult tasks in complex and unstructured environments, it becomes more of a challenge to reliably program their behavior. There are many sources of variation for a robot control program to contend with, both in the environment and in the performance characteristics of the robot hardware and sensors. Because these are often too poorly understood or too complex to be adequately managed with static behaviors hand-coded by a programmer, robotic systems some times have undesirable limitations in their robustness, efficiency, or competence. Machine learning techniques offer an attractive and innovative way to overcome these limitations. In principle, machine learning techniques can allow a robot to successfully adapt its behavior in response to changing circumstances without the intervention of a human programmer [2,3].

Classifier systems are a general-purpose inductive machine learning systems, which use genetic algorithm as the learning mechanisms.

In this paper we use genetic algorithms with reinforcement learning through the use of a classifier system. This learning mechanism ensure flexibility and adaptation to an unknown environment during a simulation.

## 2. Reinforcement Learning

Reinforcement learning is a synonym of learning by interaction. During learning, the adaptive system tries some actions (i.e., output values) on its environment, then it is reinforced by receiving a scalar evaluation (the reward) of its actions. The reinforcement learning algorithms selectively retain the outputs that maximize the received reward over time.

RL tasks are generally treated in discrete time steps. At each time step -t- the learning system receives some representation of the environment's state -s-, it takes an action -a-, and one step later it receives a scalar reward -r-, and find itself in a new state -s'-. The two basic concepts behind reinforcement learning are *trial* and *error* search and delayed.

RL can be defined as the problem faced by an agent that learns by trial and error how to act in a given environment, receiving as the only source of learning information a scalar feedback known as "reinforcement".

One key feature of RL is a trade-off between exploitation and exploration. To accumulate a lot of reward, the learning system must prefer the best experienced actions, however, it has to try (to experience) new actions in order to discover better action selections for the future [1,5].

## 3. Learning Classifier System (LCS)

A classifier system (CS) is a machine learning system that learns syntactically simple string rules, called classifiers, as introduced by Holland and Reitman (1978)[6]. These classifiers guide the system's performance in an arbitrary environment. A classifier system derives its name from its ability to learn to classify messages from the environment into general sets and is similar to a control system in many respects. As a control system uses feedback to "control" or "adapt" its output for an environment, a classifier system uses feedback to "teach" or "adapt" its classifiers for an environment.

A CS is a kind of parallel production rule system in which kinds of learning takes place. First, reinforcement learning. Second, evolutionary learning [6].

A classifier system has three major components [5]:

1. Rule and message system (performance system),
2. Apportionment of credit system, (Bucket Brigade Algorithm-BB)
3. Classifier discovery mechanisms (primarily the genetic algorithm).

### 3.1 Rule and Message System

Each classifier consists of a rule or conditional statement whose constituents are words form the ternary alphabet (0,1,#). It has one or more words or conditions as the antecedent, an action statement as the consequent, and an associated strength. The rule portion has the template:

IF<condition>& < condition2>&...& <condition N> THEN <action>

where,

<condition> is encoded as a finite-length string from the alphabet {0,1,#}

<action> is encoded as a finite-length string from the alphabet {0,1}.

The "#" symbol acts as a wild card or "don't care" in the condition, matching either a 0 or 1. This allows for more general rules. The more "don't care" symbols the more general the rule. The measure used to quantify is called: specificity. The specificity of a classifier is the number of non # symbols in the antecedent. If a classifier's antecedent consists of all # characters then the specificity is zero, if there are no # characters in the antecedent then the specificity is equal to the antecedent's string length. The strength portion of the classifier gives a measure of the rule's past performance in the environment in which it is learning. That is, the higher a classifier's strength the better it has performed and the more likely it will actually be used when the condition matches an environmental message and to reproduce when the GA is applied.

The messages, generated from the environment (or from the action of other classifiers), match the condition part of the classifier rule. Therefore, an action is a type of message, with the consequence of an action being the modification of the environment (or the attempted matching with another classifiers in some classifier systems).

The messages from the environment, which match the antecedent of the classifiers, are filtered and converted via input sensors. The sensors (called detectors in classifier system parlance) discriminately select certain aspects of the environment to sense and then translate the input to a binary form which can be processed by the classifiers [6].

The actions of the classifiers modify the environment via the effectors (or output interface). The effectors translate the binary action into a form which is appropriate to modify the environment within an envelope of allowable modifications [4].

### 3.2.Apportionment of Credit System

The apportionment of credit system deals with the modifications in strength of classifiers as the classifier system learns (Booker, et al. 1989)[6]. Strength modifications occurvia three interrelated mechanisms:

- Auction,
- Reinforcement& punishment,
- Taxation.

As the classifier system receives messages from the environment, all the classifiers which match one (or more) of the messages compete, by submitting a bid, in an auction to determine a victorious classifier that will affect the environment. The victorious classifier's modification will be beneficial or detrimental to the environment. With this feedback, the apportionment of credit system appropriately uses reinforcement and punishment to increase or decrease the strength of the victorious classifier that caused the modifications. Finally, taxation is levied on each classifier per iteration and on each classifier that submits a bid during an auction [6].

### 3.2.1.Auction: Bidding and Competition:

An auction is performed among all the classifiers, which have an antecedent that matches at least one of the environmental messages. The classifiers system's detectors receive input from the environment and assemble the input into environmental messages. Each classifier attempts to match each environmental message, with each classifier that matches bidding in the auction.

With the matching classifier pool determined, the auction commences. Each classifier participating in the auction submits a bid, the bid is a function of the classifier's strength and specificity. Only the bid of victorious classifier is paid, so only the victorious classifier has its strength decreased by the amount of its winning bid. The bid of classifier is calculated in equation (1).

$B_i = C_{bid} S_i$ ...(1)

where

$C_{bid}$ : Classifier bid coefficient: positive constant less than unity that acts as an overall risk factor influencing what proportion of a classifier's strength will be bid and possibly lost on a single step. (Usually $C_{bid} \approx 0.1$).

S : Strength and I is classifier index.

Competing classifier in above equation is not used directly to determine the auction winner, random noise is added to the auction. Therefore the effective bid, EB, is calculated as the sum of the deterministic bid, Bi, and a noise term, $N(\sigma_{bid})$ as shown in equation [2].

$EB_i = B_i + N(\sigma_{bid})$ ...(2)

Where : $\sigma_{bid}$ : Noise term coefficient: positive constant less than unity. The noise N is a function of the specified bidding noise standard deviation $\sigma_{bid}$.

### 3.2.2. Reinforcement and Punishment:

A trainer is necessary to determine whether the environmental modify-cation was beneficial or detrimental. Some machine learning systems require a tutor trainer, which knows the correct or best answer, enabling the system's actual response to be compared with the correct response. Fortunately, a classifier system requires only the more flexible reinforcement trainer.

Reinforcement learning requires only positive or negative feedback from the reinforcement trainer as a conesquen-ce of a response.

When the victorious classifier creates a beneficial effect to the environment, the trainer sends positive feedback causing an increase in the victorious classifier's strength. Conversely, a detrimental effect leads to punishment. Since the victorious classifier's strength decreases when it wins the auction and pays its bid, punishment occurs implicitly anytime a reward is not provided. In addition, an adjunct strength reduction may occur. If the trainer has the ability to rank environmental effects, then the rewards and punishments can be scaled appropriately.

The strength $S_i(t + 1)$ of a classifier I at the end of iteration is:

$$S_i(t + 1) + S_i(t) - B_i(t) + R_i(t) - T_i(t) \quad \ldots(3)$$

where,

$S_i(t)$ Strength of classifier I at beginning of iteration t.

$R_i(t)$ Reward from the environment during iteration t.

$B_i(t)$ Classifier's bid during iteration t. Only paid if victorious.

$T_i(t)$ Tax

Again, classifier I only makes a bid payment if victorious in the auction and effects the environment. The reward factor, $R_i(t)$, is only non-zero if the classifier won the auction on the previous iteration. The reward (or punishment) for the action at iteration t will not be applied until iteration t + 1. Note that $R_i(t)$ is less than zero for punishment, and greater than zero for reward. [6]

**Taxes**

Taxation occurs to prevent the classifier population from being cluttered with artificially high strength classifiers of little or no utility [6].

There are two types of taxes:

- Life tax.
- Bid tax

The life tax, Taxlife, is a fixed rate tax applied to every classifier on every iteration. The purpose is to reduce the strength of classifiers that rarely or never are matched and therefore provide little or no utility. Non-producing classifier's strengths are slowly decreased, making them candidates for replacement when the classifier discovery mechanisms (primarily the genetic algorithm) create new classifiers.

The bid tax, Taxbid, is a fixed rate tax that is applied to each classifier that bids during an iteration. One reason for a bid tax is to penalize overly general classifiers, i.e., classifiers that bid on every step but perhaps seldom win because they have a low specificity which leads to low bids and so a low chance of winning the auction to post effector messages (Riolo-1988). [6]

## 3.3. The Rule Discovery System (Genetic Algorithm)

The bucket brigade provides a clean procedure for evaluating rules and deciding among competing alternatives. Yet we still must devise a way of injecting new possibly better rules into the system. This is precisely where the G.A steps are used. Using the simple genetic algorithm new rules are created using (reproduction, crossover and mutation). These rules are then placed in the population and processed by the auction, payment, and reinforcement mechanism to properly evaluate their role in the system. [2]

After applying the GA, only some of the rules are replaced. The new rules will be tested by the combined action of the performance and credit apportionment algorithms. Because testing a rule requires many time steps, Gas are applied with a much lower frequency than the performance and apportionment of credit systems.[6]

A complete classifiers system needs some means of generating new rules for use in the performance and learning systems. Genetic algorithms

(GA) techniques have been used as the main source of rule discovery in CS.

## 4. Simulation of Robot

In this paper genetic algorithms are used to learn complex behaviors for robot using distributed learning classifier systems as a control system. To investigate this goal we proposed a simulation system called Robot-Moving Light-Goal System (RMLGS). In this system the simulation environment is a two dimensional environment where there were three objects that it could perceive. The objects are as follows:

* A moving light source, which moves randomly toward the goal and the initial position of the light is random.

* A robot, which chases a moving light to catch it before it reache the goal.

There are four probabilities for initial positions of a robot. They are as follows:

- Only one robot always has fixed initial position.
- Only one robot always has random initial position.
- Three robots have fixed initial position and one of them becomes active by control depending on the distances between a moving light and them.
- Three robots have random initial position and one of them becomes active by control depending on the distances between a moving light and them.

  - The goal, always has a fixed position.
  - Emergency, can only be heard when the distance between a moving light and the goal becomes less than or equal to the known fixed distance.

In this proposed system a distributed classifier system (DCS) uses two-level hierarchical architecture. This DCS consists of three classifier systems: two classifier systems (LCS-1 & LCS-2) in the high level, each of them learns the basic behavior – (LCS-1) learns robot to walk one step toward the light when it does not hear the sound of emergency but (LCS-2) learns robot to walk two steps toward the light when it hears the sound of emergency-, and the third classifier system (LCS-C) is in the low level, which learns to switch between the two basic behaviors.

### 4.1. The Structure of RMLGS

RMLGS is implemented as a set of three classifier systems organized in two-level hierarchical architecture: two classifier systems (LCS-1 & LCS-2) learn the basic behaviors, while one (LCS-C) learns to switch between the two basic behaviors. The structure is shown in Figure (2).

### 4.2. Representation of RMLGS

#### 4.2.1. Representation of LCS-C

- *Environmental message*

The length of messages which LCS-C receive from the environment is always 5-bits as shown in Figure (3).

- *The action:*

The action's length of LCS-C is always one bit:

0 If LCS-C sends message to LCS-1.

1 If LCS-C sends message to LCS-2.

#### 4.2.2. Representation of LCS-1 & LCS-2

- *Input message*:

The length of messages which LCS-1 or LCS-2 is received from LCS-C is always 4-bits as shown in figure (4).

- *The action*

The length of action which LCS-1 or LCS-2 is send to the environment is always 4-bits as shown in Figure (5).

## 5. The performance measure of RMLGS:

In RMLGS the following equation is used:

$$P = \frac{\text{Number of correct responses}}{\text{Total number of responses}} \qquad 0 \le P \le 1 \qquad \ldots(4)$$

as performance measure. That is, performance is measured as the ratio of correct move to total moves performed from the beginning of the simulation. The notion of "correct" response is implicit in the reinforcement program (RP): a response is correct if and only if it receives a positive reinforcement. Therefore, we call the above defined ratio the "cumulative performance measure induced by the RP".

RMLGS uses four types of environment, as follows:

1- A light in random position, and one robot in fixed position.
2- A light in random position, and one robot in random position.
3- Random position, and three robots in fixed position.
4- Random position, and three robots in random position.

With each type of the environment in RMLGS, the performance of LCS-1 is calculated independently of LCS-2 and the performance of LCS-2 is calculated independently of LCS-1, the performance of each LCS-1 and LCS-2 is 100% that means the whole system performance will be 100%.

**Notes:**

+: A moving light moves to wards the goal.

_: The robot moves one step towards a moving light by using LCS-1 (Chasing behavior).

$\wedge$: The robot moves one step towards a moving light by using LCS-2 (Approaching behavior).

## 6. Conclusion:

- Learning to perform a complex behavior becomes simple. Complex behavior is portioned into simple behaviors.
- Distributed classifier system gives the ability to build any system with complex behavior.
- Using classifier system for each simple behavior will reduce the complexity of building the system and its learning.
- Using two-level hierarchical architecture makes the robot learn the behaviors at much less time for execution.
- Using different initial states for the objects in simulation environment makes the system more flexible.
- Building a temporary memory for receiving more than one environmental message and then automatically executing them one message in each cycle makes the system more realistic.
- The capability of robot to sense in all directions makes the system more efficienty and flexible.
- Using classifier store with variable size of population, makes the system more flexible to destroy or add new rules from and to the classifier store.
- Using reinforcement learning is an easy way to notice the performance of robot that means helping the user to view the performance on the desired

environment, for example showing the report that shows all the details of each run, and makes changes if needed for new test.

- Using a reward for winner classifiers and punishment for the others. Reinforcement learning is better way than using only reward.
- Strength should be used a fitness measure to find the correct solution.

## References

1- Andres Perez "Introduction to Reinforcement Learning", 1998.

2- D.E. Goldberg "Genetic algorithms in search, optimization, and machine learning", New York, Addison wesely, 1989.

3- Francis Clark " Genetic algorithms", Genetic Algorithm. htm, 1997.

4- Luger & Stubblefield "Artificial Intelligence Structures and strategies for complex problem solving", pp.713-736, 1998.

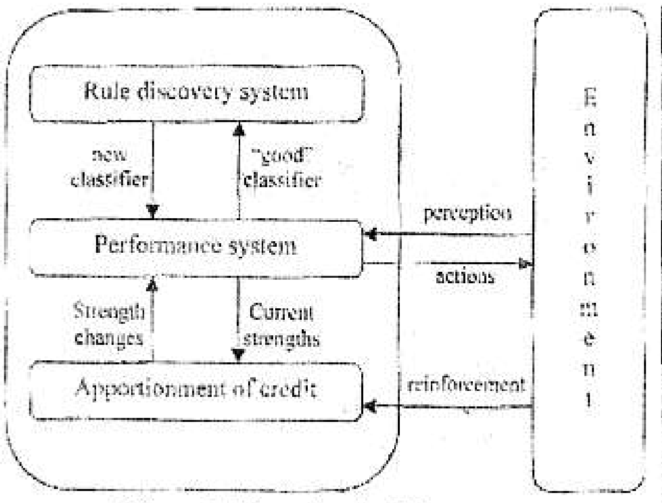
5- Marco Dorigo & Marco Colombetti The MIT Press-Massachvsetts "Robot shaping (An experiment in behavior engineering)"- 1998.

6- Robert A. Richards "Zeroth-order shape optimization utilizing a learning classifier system", 1995.

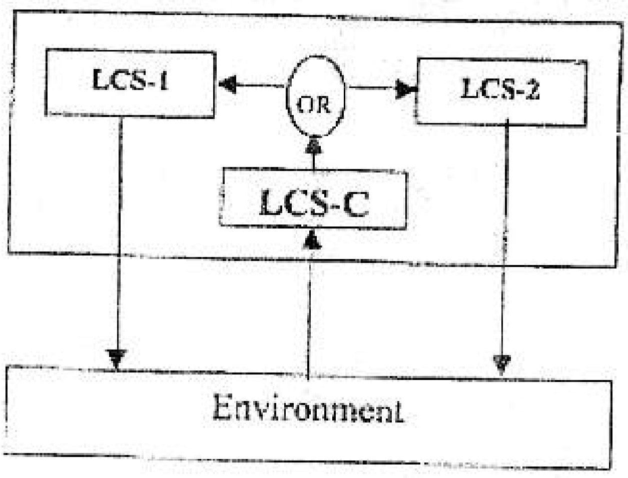
7- Stan Franklin "Artificial Minds" – MIT Press, 1997. USA

8- Stefan Schaal "Robot learning"- University of Southern California, Paper, 2001.

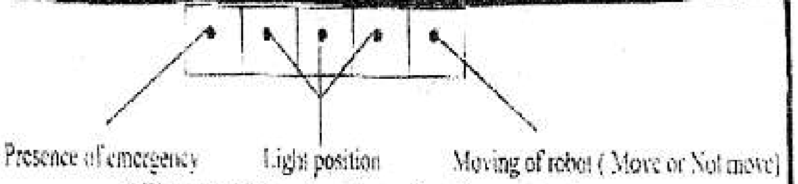
9- Yves Kodratoff and Alan Hutchinson "Machine and human learning" – 1989.



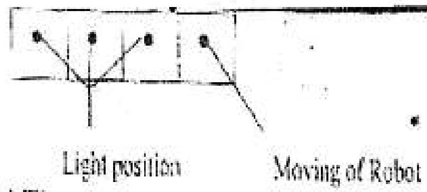
**( Figure 1 ) Learning classifier system**



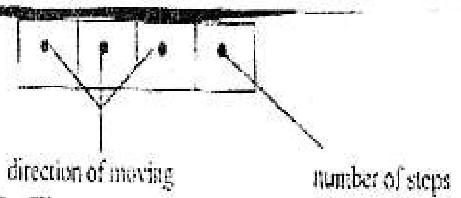
**( Figure 2 ) The structure of RMLGS**



**( Figure 3 ) The structure of environmental message.**

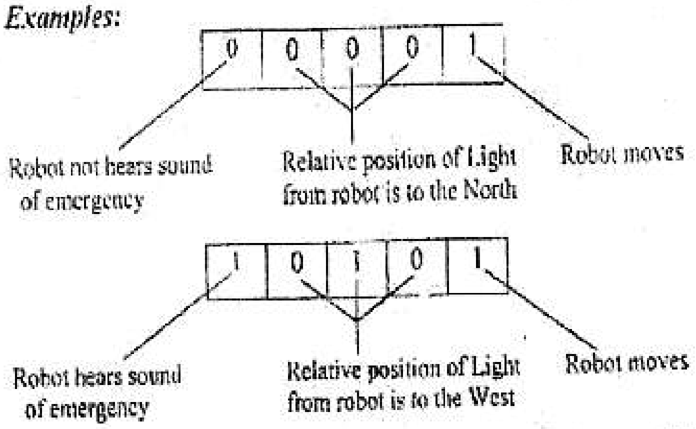


**( Figure 4 ) The structure of input message of LCS-1 & LCS-2**



**( Figure 5 ) The structure of action of LCS-1 and LCS-2 .**

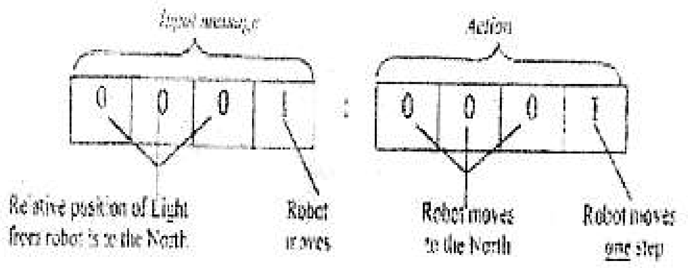
*Examples:*



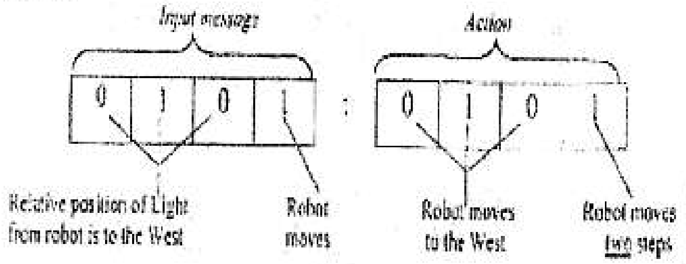
0
0
0
0
1
Robot not hears sound of emergency
Relative position of Light from robot is to the North
Robot moves
1
0
1
0
1
Robot hears sound of emergency
Relative position of Light from robot is to the West
Robot moves

*Example:*

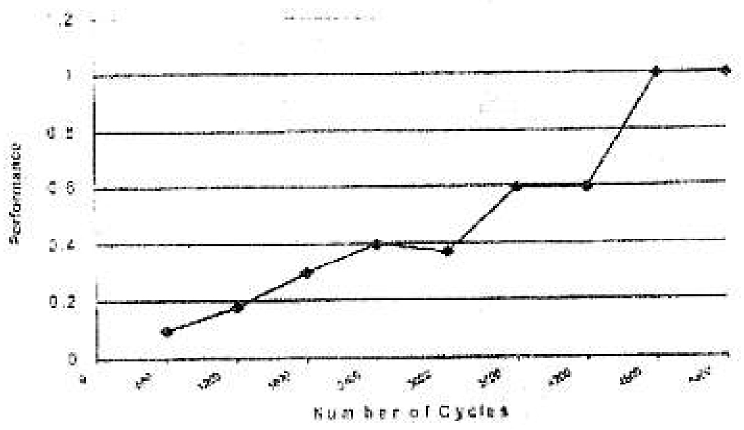
1) With LCS-1



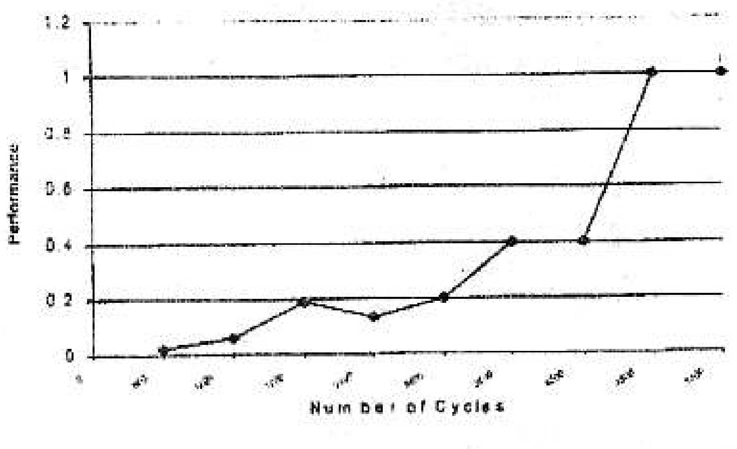
2) With LCS-2



The curve of performance for LCS-1 & LCS-2 will be shown in figures nd 7) as the following:

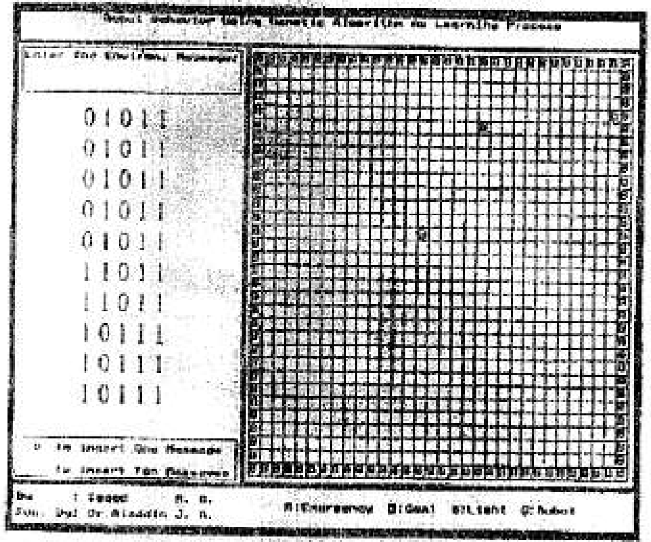


( Figure 6 ) Proportion of correct trails from start [LCS-1 & LCS-2]



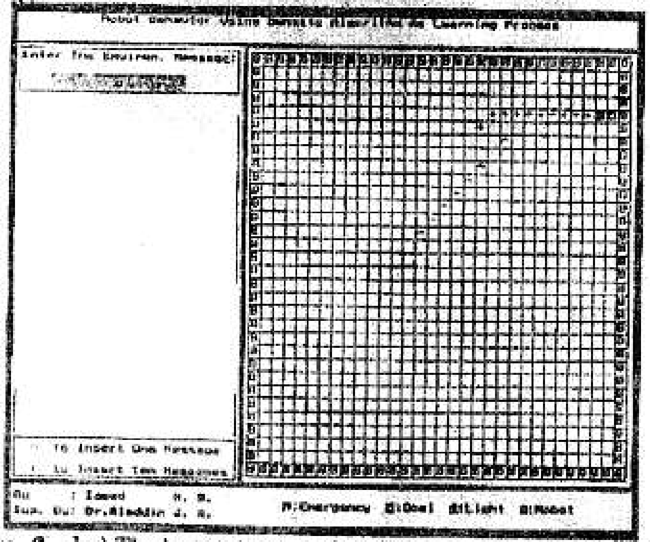
( Figure 7) Proportion of correct trails (last fifty)[LCS-1 & LCS-2]

Note:
- : A moving light moving towards the goal.
- : The robot moving one step towards a moving light by using LCS-1 (Chasing behavior).
- : The robot moving two steps towards a moving light by using LCS-2 (Approaching behavior).



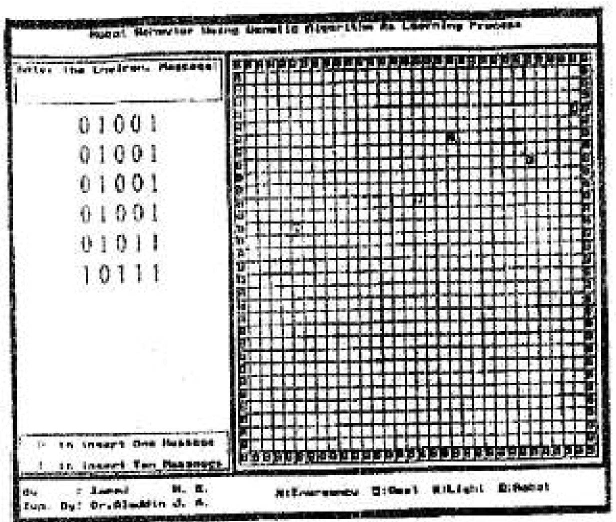
```
Robot Behavior Using Genetic Algorithm As Learning Process

Enter The Environ. Messages

01011
01011
01011
01011
01011
11011
11011
10111
10111
10111

To Insert One Message
To Insert Ten Messages

By      : Taeed        H. H.
Sup. By: Dr.Alaadin J. H.        E:Emergency  G:Goal  L:Light  R:Robot
```

( Figure 9 -a- ) The initial state when choosing one robot in fixed position



```
Robot Behavior Using Genetic Algorithm As Learning Process

Enter The Environ. Messages

To Insert One Message
To Insert Ten Messages

By      : Taeed        H. H.
Sup. By: Dr.Alaadin J. H.        E:Emergency  G:Goal  L:Light  R:Robot
```

Figure 9 -b- ) The last state when choosing one robot in fixed position



Robot Behavior Using Genetic Algorithm As Learning Process

Enter The Environ. Message:

```
01001
01001
01001
01001
01011
10111
```

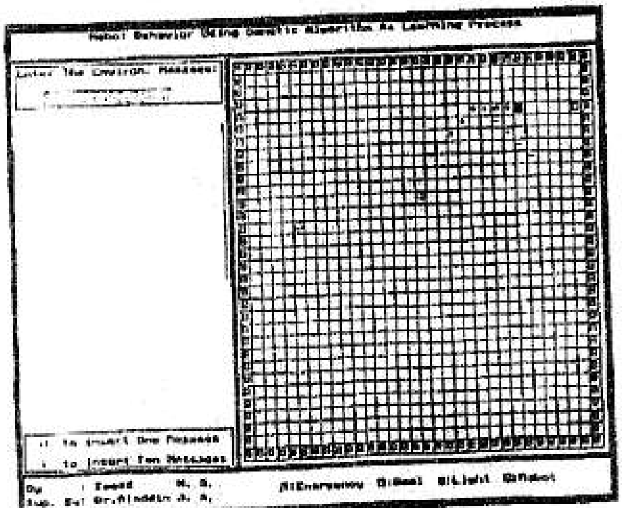
[illegible] to Insert One Message

[illegible] to Insert Ten Messages

By : Zaeed H. S.
Sup. By: Dr.Aladdin J. A.

E:Emergency G:Goal L:Light R:Robot

( Figure 8-a-) The initial state when choosing three robot in random positions



Robot Behavior Using Genetic Algorithm As Learning Process

Enter The Environ. Message:

[illegible] to Insert One Message

[illegible] to Insert Ten Messages

By : Zaeed H. S.
Sup. By: Dr.Aladdin J. A.

E:Emergency G:Goal L:Light R:Robot

( Figure 8-b-) The last state when choosing three robot in random positions