A New Hillclimber for Classifier Systems

Conference Paper · September 1997

DOI: 10.1049/cp:19971162 · Source: IEEE Xplore

2 authors:

Kwok Ching Tsui
Hong Kong Baptist University
48 PUBLICATIONS **726** CITATIONS
SEE PROFILE

Mark D. Plumbley
University of Surrey
406 PUBLICATIONS **9,004** CITATIONS
SEE PROFILE

A New Hillclimber for Classifier Systems

Kwok Ching Tsui[†,*] and Mark Plumbley[‡]

[†]Department of Computer Science
[‡]Department of Electrical and Electronic Engineering
King's College London, London WC2R 2LS, UK

Abstract

Multi-state artificial environments such as mazes represent a class of tasks that can be solved by many different multi-step methods. When different rewards are available in different places of the maze, a problem solver is required to evaluate different positions effectively and remembers the best one. A new hillclimbing strategy for the Michigan style classifier system is suggested which is able to find the shortest path and discarding sub-optimal solutions. Knowledge reuse is also shown to be possible.

1 Introduction

Classifier Systems (CSs) have been used to study complex the emergent behaviour of artificial creatures in simulated environments [2], commonly using robots both real and simulated [1, 7]. The seminal work by Holland and Reitman [3] used a simulated creature in the context of a one dimensional maze. Others [4, 6] have used a multi-state environment where a robot is required to transfer itself from one state to another to test or demonstrate various behaviours of CSs.

Maze negotiation is a class of tasks that can be attempted by many different multi-step methods. A graph can be constructed with the vertices being all the possibles states and the edges being the transitions between the states. Completing a task from beginning to end is like traversing through the multi-state of this graph.

Another characteristic of maze negotiation is that the result (success of failure) will not be known to the problem solver until the last step has been applied. Not much feedback can be obtained during the course of problem solving before the goal is found. This is analogous to chess playing where nobody can tell for sure which side will win until the end of the game. Moreover, it is difficult to decide whether an intermediate move is beneficial or not. Therefore, a system which can automatically evaluate alternatives and choose the best course of action will be of great value.

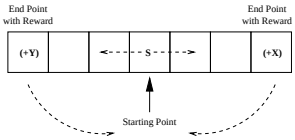
This article describes a new classifier system called *Maze Classifier System* (MaCS) which makes decisions for the robot that runs around in the mazes. Advanced features such as cover detector and cover effector operators [5] have been incorporated. A new hillclimbing mechanism which preserve intermediate solutions while searching for new ones is discussed in detail together with the ability of *MaCS* to reuse learned knowledge.

2 The Mazes

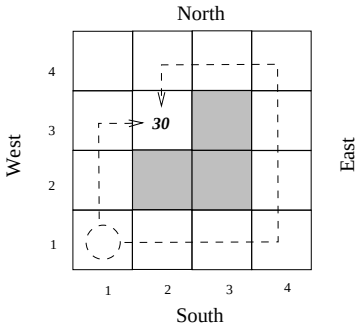
2.1 One-dimensional Maze

The one-dimensional maze used consists of a number of blocks put horizontally next to each other. Each block has a unique *identification code*. A block in the middle of the maze is designated the starting point and the blocks at both ends the end points (goals). Figure 1a is a 1-D maze with 7 states. It has a starting point 'S' and reward/penalty at each end point. In this case, two end points have rewards +X and +Y. Once an end point has been reached, the search task is completed. Whenever either of the end points is reached, whatever reward is contained in that end point is passed back to the classifier system as a reinforcement signal. The system will then be transferred to the starting point to get ready for the next run.

*current address: BT Laboratories, Martlesham Heath, Ipswich IP5 7RE, UK email: tsuikc@info.bt.co.uk



(a) A one dimensional maze with 7 states



(b) 2-D maze with obstacles

Figure 1: The mazes. The possible moves in a 1-D maze is either left or right while there are four possible directions to move in a 2-D maze.

2.2 Two-dimensional Maze

Two dimensional mazes are more difficult versions of their one dimensional counterparts. In the sample 2-D maze shown in figure 1b, the different states are labelled 1 to 4 from left to right and also from bottom to top. The position of *MaCS* is indicated by the coordinate on the maze such as (1,1) in figure 1b. There are four possible directions *MaCS* can move, namely north, south, east or west. However, it cannot walk outside the boundaries. The combinations of moves is thus much more complicated than in the case of a 1-D maze (left or right only). The shaded squares in the maze represents the obstacles over which the robot cannot walk. If the artificial creature is positioned at (1,1) and the reward of value 30 is positioned at (2,3). It should be noted that the term end-point has been extended to refer to the maze position where a reward is situated. Two possible paths that trail along the obstacles in different directions is shown by the dotted lines.

3 The Basic Classifier System

MaCS resembles a Michigan classifier system. It contains rule-like structure (classifiers) with multiple antecedents (conditions) and a single consequence (action) directed towards the environment. Associated with every state (position) in the maze is a unique message which will be made known to the classifier system via message from the detectors. Credit assignment is by the bucket bridge algorithm. Rule discovery is by a genetic algorithm. A new classifier is created whenever no messages match the classifiers using the cover detector operator. Likewise, the cover effector operator also generates a new classifier when the action of the winning classifier causes an error such as walking into the wall.

Unless specifically stated, there are 60 classifiers in the system all starting with an initial strength of 50. GA is applied every 50 generations. During bidding, each matched classifier uses 10% of its strength to participate in a noisy bidding. A bid tax and existence tax of 1% and 0.5% respectively are levied on each classifier. All the experimental results reported below are the average of 5 runs using different random seeds.

Three messages are issued by the system in each generation which encode the identification code of the block in which the system is currently situated, plus two more messages giving the identification codes of the two previous blocks it has just visited. All detector messages are prefixed by a *tag* to distinguish between the various kinds of messages. Accordingly, all of the classifiers in the classifier system have three conditions and an action.

Figure 2 shows the trajectory of a typical run along the maze using a classifier system with cover detector operator applied. The 1-D maze has a reward of 36 in state 7 while the other end (state 1) has no reward. State 4 is the starting point and whenever either of the end points is reached, state 4 becomes the current position immediately. Therefore, the desire pattern is continuous positioning at states 5, 6 and 7. *MaCS* managed to find a good set of classifiers (rules) for negotiating the maze after around 900 generations.

4 Finding Alternatives Using Hillclimbing

When solving a problem that requires multiple steps, a classifier system refines the rule chain by replacing one or more classifiers in the chain with some alternatives. This is achieved with the help of a GA which

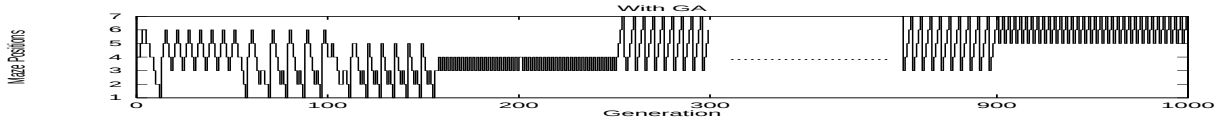


Figure 2: State transition diagram of *MaCS* using cover detector operator

provides some new classifiers. However, in two scenarios to be described below, a classifier system was not able to pick up the correct classifiers.

In a situation where either end of the 1-D maze provides some reward but of different magnitude, it is sometimes preferable to go for higher reward. The first experiment tested a classifier system on a 7-state 1-D maze with rewards of 30 and 50 at state 1 and state 7 respectively.

Consider another situation where alternative solutions sometimes require different number of steps. A better solution might mean getting more reward using less time. In the 1-D maze with 8 states, state 4 remains the starting point of the maze. Rewards at both end points (state 1 and 8) are set at the same value of +30. The preferred path is the shorter path, that is from state 4 to state 1. Varying slightly the above setup by giving a reward of +50 instead of +30 yeilds the third challenge to *MaCS*. The desired path becomes state 8 and the reward becomes more preferable than the path length.

The results in figure 3a show that *MaCS* has got stuck in a sub-optimal solution. Runs repeated using the same setup show an equal split between state 1 and 7 as the target for reward. *MaCS* tend to stick to a solution and keep reinforcing (increase the strength of) the classifiers that contribute to a reward disregarding the optimality of the solution. There is little chance for an alternative to be evaluated. A similar situation is observed in the case with eight states and unequal rewards at the end-points (figure 3c).

Out of the five independent runs for the experiment with a +30 reward at both end-points, all of them successfully reached state 1. A close look at the result of one of these runs shown in figure 3b reveals that state 1 has been reached twice before generation 600. Then between generation 600 and 800 there is a sub-optimal solution which involves states 5, 6 & 7 before reaching the desired behaviour of touring between states 1-4. Generation 600, when GA is applied, seems to be a watershed before which is a gradual move away states 1-4. It is interesting to explore a way to smoothen the change.

In an effort to rectify the problem, a new *hill-climbing* feature for *MaCS* was investigated. When a solution is found, the hillclimbing strategy works as follows:

1. update strength of the classifiers
2. calculate the *value* of the solution as

$$\frac{reward}{a + b * steps.} \tag{1}$$

   The value of a and b is taken as 1 and 0.01 respectively based on empirical results.

3. if the value of the new solution is higher than the current best, make the new solution the current best solution and *protect* the chain from being replaced (can still reproduce) *otherwise* multiply the reward just received by -1 (to make it a penalty)
4. *unprotect* the old solution
5. proceed to normal *MaCS* processing cycle until the stopping criteria is satisfied

Apart from temporary ban on competing, a new *protection* mechanism deprive the classifiers contributing to the current best solution from the action of GA. This includes the possibility of being replaced by new classifiers and reproduction. This is to ensure its existence in the classifier pool without exerting influence on the search that follows. These classifiers also receive exemption from the classifier tax system to avoid unnecessary strength decay.

In the absence of a valid solution for a set period of time, despite the help of cover operators, *MaCS* will restart the run (from the starting point). This is to eliminate lengthy but unfruitful search.

The 'compare-save-restart' hillclimbing strategy virtually performs multiple classifier runs. A gradually increasing *baseline* is formed by the reward of the current best solution since only solutions with better value will be accepted. Therefore, the baseline effectively helps to direct the search away from local minimums. The major differences between a conventional classifier system and *MaCS* with hillclimbing are:

1. *Reference Model*. Every successful search results in a useful list of classifiers that have achieved a reward. This can be valuable when a GA is used to inject new alternatives, if these useful classifiers are allowed to participate. Slight changes

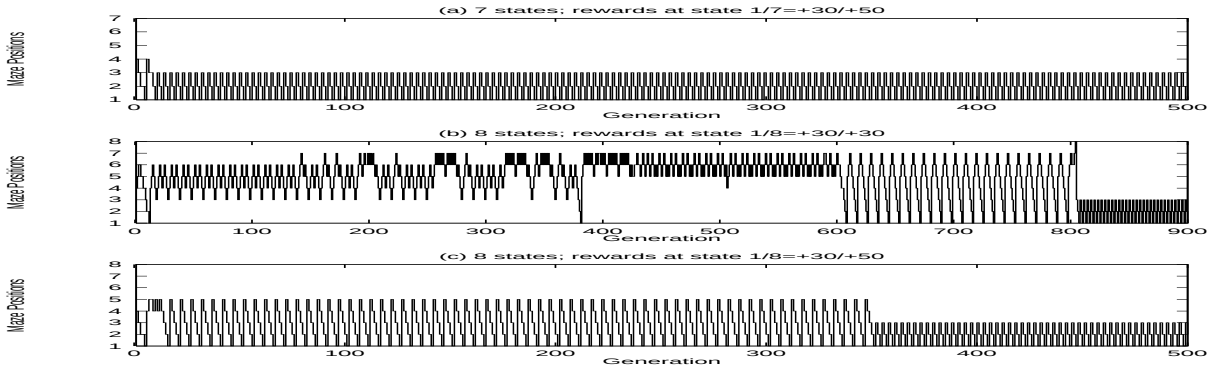


Figure 3: Results of experiments without hillclimbing

to some of the classifiers can result in a totally different classifier set. These classifiers effectively become a reference model for new classifiers (and hence potential solutions) in the generations that follow, helping the classifier system to exploit the solution at hand. It should be noted that the 'reference model' formed in this way is only in effect when a better solution is found and it is free to compete and reproduce.

Using the biggest value so far (considering reward and step size at the same time) as a baseline for accepting new solutions is another evidence of the role as a reference model in the hillclimbing strategy.

2. *Memory.* Retaining a solution as a reference model has the effect of providing a short-term memory structure for *MaCS*. This gives the classifiers in the solution resistance against being destroyed by chance. Hence, once a solution has been found, *MaCS* is always able to produce a solution. This short term memory will only be erased when its place is needed by a newer (and better) solution.

3. *Multiple Evaluation Criterion.* A solution is evaluated not just by the reward but also by the number of steps used to find the solution. This enables *MaCS* to show preference for a short path to a solution.

The hillclimbing strategy invites the criticism that it has abandoned the 'classifier economy' which was designed to sieve out the weak classifiers from the strong (and useful) ones. In reality, hillclimbing allows sub-optimal solutions to be excluded from the competition, leaving the remaining classifiers to compete. The 'classifier economy' maintained by the bucket brigade algorithm then comes into play until another sub-optimal solution is found. Therefore, instead of hindering each other, the hillclimbing strategy and the 'classifier economy' complement each other.

5 Experimental Results

Having designed the hillclimbing strategy, the following experiments were run with hillclimbing added to *MaCS*. $Reward/(1+0.01*steps)$ is used to determine the value of a solution based on the above analysis.

5.1 Simple 1-D Maze

In light of the new hillclimbing strategy for *MaCS*, the experiments described in the last section were repeated with hillclimbing and the results are given in figure 4. The results show that *MaCS* successfully found the paths to the reward in cases (unequal reward at end-points) where it failed previously. In the case of testing the ability of *MaCS* to choose a shorter path to reward, hillclimbing increases the chance of *MaCS* visiting the desirable side of the 1-D maze which contains state 1-4..

5.2 2-D Maze with Obstacles

To further test the ability of evaluating alternatives and choosing the best one, the 2-D maze shown in figure 1 which contains obstacles in the middle of the maze is used in this experiment. The two possible paths are of different lengths.

The first five rows of table 1 contains a summary of the results averaged over five runs. Within the 2,000 generations in each run, *MaCS* needs an average of 100 generations to locate the first solution and about 60 ± 46 solutions are found. Among the solutions, there are an average of 32 ± 43 good solutions (the shorter path) and an average of 3.4 bad solutions (the longer path). This represents a substantiate bias towards the better solution.

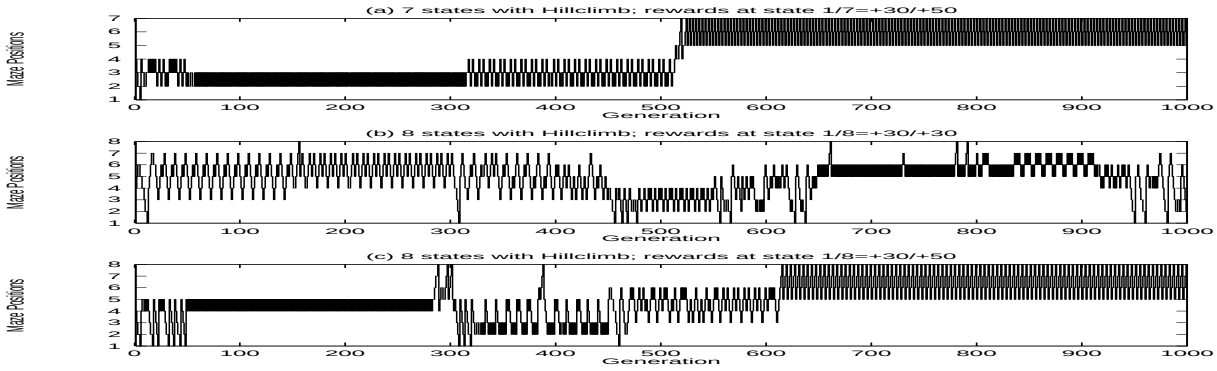


Figure 4: Results of experiments with Hillclimb)

average of 5 runs	First solution	$\overline{x} = 70, \sigma = 70$
	First good solution	$\overline{x} = 267, \sigma = 338$
	no. of solution	$\overline{x} = 59, \sigma = 46$
	no. of good solution	$\overline{x} = 32, \sigma = 43$
	no. of bad solution	$\overline{x} = 3.4, \sigma = 2.5$
overall average	gen. between solutions	$\overline{x} = 30, \sigma = 60$
	gen. between good solutions	$\overline{x} = 52, \sigma = 138$

Table 1: Summary of results for negotiating a maze with obstacles

The results in the last two rows of table 1 show that the average time *MaCS* requires, after restart, to reach a good solution is twice of that required to locate any solution. This is an evident of exploration into unknown areas in the solution space by *MaCS*. The next experiment tackles the generalisation ability of *MaCS* by making use of a solution to solve similar problems.

5.3 Search Through Novel Situations After Learning

Using the final classifier set from the experiment in section 5.2 as the initial classifier set, *MaCS* was run with a new starting point. The genetic algorithm was switched off to prevent the classifiers from being destroyed and new classifiers being introduced. Control runs were conducted using a random initial population and a genetic algorithm for comparison.

Two new starting points of (4,1) and (1,4) were tried. Two optimal, equidistant paths can be identified in both cases but only one of them has been learned. Starting off at (1,4) and reaching (2,3) means approaching the goal state from a totally different direction as before. Prior knowledge is considered not very helpful in this case. In contrast, starting off at (4,1) has a longer path to the reward. One of the optimal paths has half of its length on the learned path. Provided the route from (1,1) to (2,3) has already been captured, *MaCS* just has to concentrate on finding the route from (1,4) to (1,1). However, notice that the existence of prior knowledge is unknown to the algorithm *MaCS* represents. Learned knowledge becomes useful only when the appropriate circumstances arise.

Table 2 shows the results of this experiment averaged over five runs. When starting at (1,4), the presence of prior knowledge helps in finding the first solution but not the first good solution. This indicates that prior knowledge does not help. This is probably due to the short distance between the starting point and the end point.

When starting at (4,1), *MaCS* requires about half the time to find the first good solution in the presence of prior knowledge. Prior knowledge has significantly improved the performance of *MaCS*.

6 Conclusion

This article has shown the development of a classifier system called *MaCS* for negotiating one-dimensional and two-dimensional mazes. *MaCS* was developed primarily to show the problem solving ability of a classifier system in situations where there might exist more than one optimal solution. The ability to evaluate various alternative solutions is made possible by a new *hillclimbing* strategy. The hillclimbing strategy

Prior Knowledge?	Not Require		Require	
	Yes	No	Yes	No
First solution	$\overline{x} = 16$ $\sigma = 13$	$\overline{x} = 99$ $\sigma = 85$	$\overline{x} = 60$ $\sigma = 57$	$\overline{x} = 240$ $\sigma = 186$
First good solution	$\overline{x} = 328$ $\sigma = 392$	$\overline{x} = 281$ $\sigma = 354$	$\overline{x} = 437$ $\sigma = 322$	$\overline{x} = 982$ $\sigma = 562$

Table 2: Summary of results for search in a novel environment after learning

empowers *MaCS* with memory of solutions already found, a reference model to develop the classifiers for further exploration, and a bias towards shorter path to a reward by assigning *value* to different solutions which is a function of reward and path length. Both the one dimensional and two dimensional maze experiments showed the problem solving power of *MaCS*, even a more difficult situation where there are more feasible solutions. The presence of obstacles in the two dimensional maze is even harder than usual.

The cover operators incorporated into *MaCS* provides a powerful classifier generation mechanism at times when no classifier matches the detector messages or when inappropriate actions were suggested. Replacement of weak classifiers when applying the cover operators speeds up the process of artificial evolution. The cover effector operator also penalises those classifiers that have suggested action inappropriate to the situation by imposing bid tax on it. On the whole, the cover operators are beneficial to *MaCS* and may replace part of the job of a genetic algorithm.

A closer look at the details of the results reveals that only some of the classifiers that have learned the path from previous runs were among the classifier chain of the first optimal solution. Despite the good results, *MaCS* seem not to be able to make full use of any learned knowledge.

To further encourage the reuse of learned knowledge, a *macro* operator is under development. Macros allow *MaCS* to search for useful classifier sequence in addition to individual classifiers. They are created by randomly choosing few classifiers from the successful classifier chain to form a new classifier. The only difference between a macro and a normal classifier in the presence of more than one action. This calls for modification to the structure of *MaCS* to accommodate variable length classifiers. An possible application of *MaCS* which is under investigation is for optimisation of neural network architecture.

7 Acknowledgments

K.C. Tsui is partly supported by the Rotary Foundation through the provision of 1994-1995 Rotary Ambassadorial Scholarship. Dr. Plumbley is partly supported by grant (GR/J383987) from the UK Science and Engineering Research Council.

References

[1] Marco Dorigo and Marco Colombetti. Robot shaping: Developing situated agents through learning. Technical Report TR-92-040, International Computer Science Institute, 1993.

[2] J. H. Holland. *Adaptation in Natural and Artificial Systems.* MIT Press, 2nd edition, 1992.

[3] John H. Holland and Judith S. Reitman. *Cognitive Systems Based on Adaptive Algorithms*, pages 313–329. Academic Press, Inc., 1978.

[4] Rick L. Riolo. Bucket brigade performance: I. Long sequence of classifiers. In John J. Grefenstette, editor, *Genetic Algorithms and their Applications : Proceedings of the Second International Conference on Genetic Algorithms*, pages 184–195. L. Erlbaum Associates: Hillsdale, N.J., 1987.

[5] Rick L. Riolo. The emergence of coupled sequences of classifiers. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 256–264. Morgan Kaufmann: San Mateo, Calif., 1989.

[6] T. H. Westerdale. A defense of the bucket brigade. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 282–290. Morgan Kaufmann: San Mateo, Calif., 1989.

[7] Stewart W. Wilson. Classifier systems and the animat problem. *Machine Learning*, 2:199–228, 1987.