Behavior Learning in Minority Games

Guanyi Li, Ying Ma, Yingsai Dong, and Zengchang Qin*

Intelligent Computing and Machine Learning Lab
School of Automation Science and Electrical Engineering
Beihang University, Beijing, 100191, China
zcqin@buaa.edu.cn

Abstract. The Minority Game (MG) is a simple model for the collective behavior of agents in an idealized situation where they have to compete through adaptation for a finite resource. It has been regarded as an interesting complex dynamical disordered system from a statistical mechanics point of view. In this paper we have investigated the problem of learning the agent behaviors in the minority game. We assume the existence of one "intelligent agent" who can learn from other agent behaviors. We consider two scenarios in this research: (1) Given an environment with complete information, i.e., all records of agents' choices are known to public. The intelligent agent can use a Decision Tree to learn the patterns of other agents and make predictions. (2) If we only know the data of collective behaviors, we assume the data are generated from combining the behaviors of variant groups of agents. The intelligent agent can use a Genetic Algorithm to optimize these group parameters in order to get the best guess of the original system. The experimental results show that, in this configuration of MG in both environments with complete information and incomplete information, the intelligent agent can learn from history data and predict which side is the minority.

1 Introduction

Agent-based experimental games have attracted much attention among scientists in different research areas, particularly, psychology [13], economics [10,14] and financial market modeling [5,7,10]. Agent-based models of complex adaptive systems (CAS) provide invaluable insight into the highly non-trivial collective behavior of a population of competing agents. These systems are universal and we aim to model the systems where involving agents are with similar capability competing for a limited resource. Agents share global information, which is in turn generated by the action of the agents themselves, and they learn from past experience.

The field of econophysics [9] is an area that such complex system models may be applicable: every agent knows the history data in a stock market and must decide how to trade based on this global information. Among these models, minority game (MG) [4] is an important model in which an odd number N of agents successively compete to be in the minority. It can be regarded as a simplified version

* Corresponding author.

of EI Farol bar problem [1], in which a number of people decide weekly whether go to the EI Farol bar to enjoy live music in the risk of staying in a crowd place or stay at home. As a new tool for learning complex adaptive systems, the minority game has been applied to variety areas [7,8] especially in financial market modeling [2,3]. But this model has limitations when describing the real case where people try to make a better choice. The first thing we notice is that not all agents use the same pattern to select strategy. In real-life scenarios, the complexity of marketing world is embodied in existence of varieties types of agents using strategies based on their own rules. Several patterns of agents should have existed such as agents who make random decisions. Also it is unrealistic to have a population with all agents following the rules of a game. There must be some more intelligent ones who make a more wise decision by learning from others.

The previous unrealistic assumptions are not enough for us to have a good understanding of the dynamics of an adaptive system (e.g., a market). In this paper, we propose a model that efficiently figures out limitations of previous models. Through observing the dynamics, we continuously increase complexity of our model by adding diverse types of agents in order to simulate and analyze intricate systems. We also introduce intelligent agents that can learn adaptively by using machine learning and data mining. We consider two learning scenarios in this paper: (1) learning with *complete information* that includes all records of other agents' choices and winning outcomes at each time and (2) learning with *incomplete information* that only contains winning outcomes. Eventually, the intelligent agent can analyze and estimate the environment to maximize own profits in terms of winning probability. It is a new way to understand the relationship of micro-behaviors and macro-behaviors by utilizing minority game model and machine learning.

This paper is organized as the following: section 2 introduces the minority game and propose the framework in which an intelligent agent can learn the patterns of other agents by using complete information. In section 3, we assume an environment in which the intelligent agent does not have the complete information. We propose a method of using a genetic algorithm for discovering the composition of agents in order to predict the macro-behavior of the MG. By conducting a series of experiments, we verify the usefulness of this learning mechanism and conclusions are given at the end.

2 Learning in the Minority Games

In the minority game, there is an odd number of players and each must choose one of two choices independently at each turn. The players who end up on the minority side win. It is originated from the El Farol Bar problem that was originally formulated to analyze a decision-making method other than deductive rationality, the minority game examines the characteristic of the game that no single deterministic strategy may be adopted by all participants in equilibrium [15]. In this section, we will set an environment that is populated by a few diverse groups of agents. We aim to design one type of intelligent agents that can learn from the patterns of other agents and make choices in order to be on the minority side.

2.1 Strategies of Agents

Formally, the minority game consists of N (an odd number) agents, at time t ($t = 1, \ldots, T$), each agent need to take an action $a_i(t)$ for $i = 1, \cdots, N$, to attend room A or B.

$$a_i(t) = \begin{cases} A & \text{Agent i choose room A} \\ B & \text{Agent i choose room B} \end{cases} \tag{1}$$

In each round of the game, the agents belonging to the minority group are the winners. The winning outcome can be considered as a binary function $w(t)$. Without loss of generality (w.l.o.g), if A is the minority side, we define that the winning outcome is 0, otherwise, it is 1. In this paper, the winning outcomes are known to all agents.

$$w(t) = \begin{cases} 0 & \#(a_i(t) = A)|_{i=1,\ldots,N} \leq (N-1)/2 \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

where $\#()$ is the function for counting numbers: for all the agents (i runs from 1 to N), if the number of agents that satisfy the condition $a_i(t) = A$ is less than or equal to $(N-1)/2$, then $w(t) = 0$; otherwise, it is $w(t) = 1$. We assume that agents make choices based on the most recent m winning outcomes $h(t)$, which is called *memory* and m is called the *length of memory.*

$$h(t) = [w(t-m), \ldots, w(t-2), w(t-1)] \tag{3}$$

Given the outcome $w(t)$ at the moment t, agent i keeps a record $r_i(t)$ that tells whether it has won the game or not.

$$r_i(t) = \begin{cases} Win & \text{Agent i wins at time t} \\ Loss & \text{Agent i loses at time t} \end{cases} \tag{4}$$

In minority game, we usually assume that each agent's reaction based on the previous data is governed by a "strategy" [2,3,4]. Each strategy is based on the past m-bit memory which are described as a binary sequence. Every possible m-bit memory are mapped in correspond to a prediction of choosing room A or B in the next round. Therefore, there are 2^{2^m} possible strategies in the strategy space. Agents in the same fixed strategy group share one strategy randomly selected from the strategy space. Given the memory $h(t)$, the choice for the agent i guided by the strategy S is denoted by $S(h(t))$. Table 1 shows one possible strategy with $m = 3$. For example, $h(t) = [000]$ represents that if the agent who choose A in the latest three time steps win, the next round (at time t) choice for this agent will be $S([000]) = A$. A strategy can be regarded as a particular set of decisions on the permutations of previous winning outcomes.

Giving the history data of all winning outcomes, training data for an agent can be sampled by a sliding window with size m on each time step. At time t,

Table 1. One possible strategy with $m = 3$, the current choice of agent is decided by its previous 3 step memory

$h(t)$	000	001	010	011	100	101	110	111
$S(h(t))$	A	B	B	A	B	A	B	B

the target value (either A or B) is the agent's actual choice at the current time. Therefore, training set for agent i can be formally defined as:

$$\mathcal{D}_i = \{(h(t), a_i(t))\} \qquad for \quad t = 1, \dots, T \tag{5}$$

The left-hand side figure of Fig. 1 shows a schematic illustration of how to sample the training data with $m = 3$.

2.2 Decision Tree Learning

Decision tree induction is one of the simplest and yet most successful learning algorithms. It has been widely used in numerous machine applications for its simplicity and effectiveness [12]. A decision tree can be regarded as a set of rules which are interpreted from branches of the tree. In this paper, the tree model we use is actually probability estimation tree [11] because we are considering the probability distribution over two possible choices of A and B.

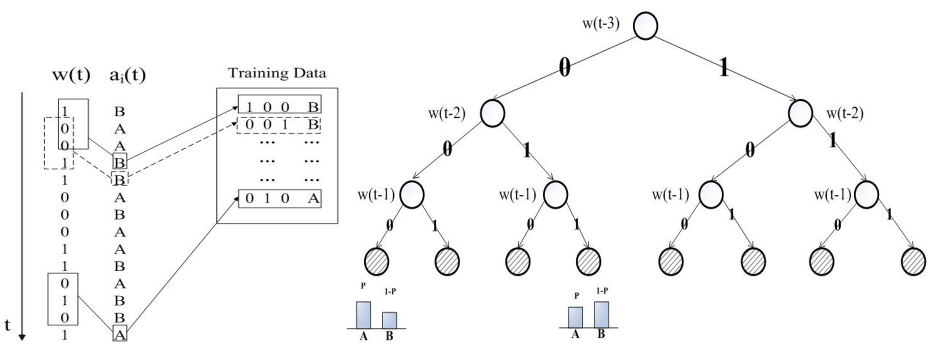

Fig. 1. Left-hand side: training data $\mathcal{D}$ is obtained from a sliding window of size m. Right-hand side: A schematic decision tree trained on the database given on the left.

The right-hand side figure of Fig. 1 shows a decision tree for modeling behaviors of agent i based on the training date given on the left. For each branch $W = [w(t-m), \dots, w(t-2), w(t-1)]$, there is an associated probability distribution on possible agent's choices (i.e., A or B) that is calculated based on the proportion of these two kinds of data falling through the branch. Or formally:

$$P(A|W) = \frac{\#(h(t) = W \wedge a_i(t) = A)}{\#(h(t) = W)}\Bigg|_{(h(t),a_i(t))\in\mathcal{D}_i} \tag{6}$$

where #() is the counting function in eq. 2 and eq. 6 means that: given a branch, it looks for all matching string from the training data and checks how many of them chose A. If agent i follows a particular fixed strategy and game is repeated for n ($n \gg 0$) times. We then can get a fair good estimation of the behaviors of agent i given the 3-step memory training data.

2.3 Learning with Fixed Strategies

Some previous studies on MG or other multi-agent systems usually assume agents are homogeneous [4,5,10]. The first thing we notice is that not all agents use the same pattern to select strategy. In real-life scenarios, the complexity of marketing world is embodied in existence of varieties types of agents using strategies based on their own rules. In the following experiment, N agents are divided into four groups. The agents in first group make random choices between A and B using an uniform distribution, which are called *random agents*. This is a realistic assumption in the real-world: there is always something without patterns that we cannot control or even understand. The second and the third group of agents follow two particular strategies S_1 and S_2, respectively (see table 2). The remaining one agent is a special agent called the "intelligent agent" who can observe not only the history of winning, but also all the records of each agent's choices. Based on these data, the intelligent agent can use decision tree (or other machine learning models) to learn the patterns of each agent. His decision making is then based on its predictions on the other agents.

Table 2. Two fixed strategies with $m = 4$ are used in experiments

$h(t)$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
$S_1(h(t))$	A	A	B	B	B	A	B	B	B	A	A	A	A	A	B	B
$S_2(h(t))$	A	B	B	B	A	A	A	B	A	A	B	A	B	B	A	B

For each agent i, its current choice $a_i(t)$ can be predicted by the decision tree learning based on the training data (see eq. 5). At time t, the probability that the intelligent agent choose A, $P_I(A)$, is calculated based on its estimation of other agents' choices $a_i(t)$ where $i = 1, \ldots, N$.

$$P_I(A) = 1 - \left.\frac{\#(a_i(t) = A)}{\#(a_i(t) = A) + \#(a_i(t) = B)}\right|_{i=1,\ldots,N} \tag{7}$$

and $P_I(B) = 1 - P_I(A)$. The above equation can be interpreted that the intelligent agent will go to the room that most of agents won't go based on its predictions. Simply, the intelligent agent makes choice based on its estimation $P_I(A)$ and $P_I(B)$, or formally:

$$a_I(t) = \begin{cases} A & P_I(A) > P_I(B) \\ B & \text{otherwise} \end{cases} \tag{8}$$

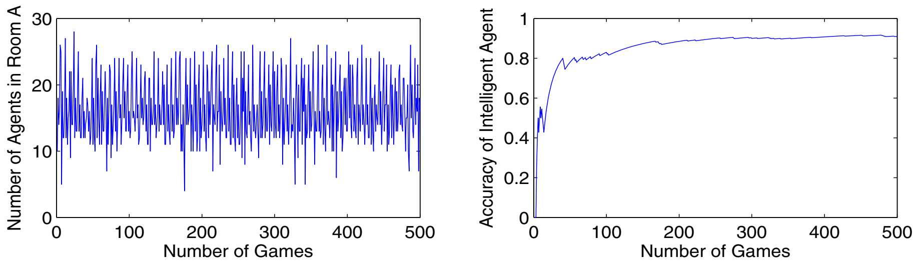

Fig. 2. Left-hand side: the number of agents in room A in 500 runs of fixed strategies MG experiments. Right-hand side: the accuracy of the intelligent agent.

The accuracy of wining for the intelligent agent I can be obtained by:

$$AC_I(t) = \left.\frac{\#(r_I(t) = Win)}{\#(r_I(t) = Win) + \#(r_I(t) = Loss)}\right|_{t=1,\ldots,T} \tag{9}$$

In the following experiment, we set the total number of agents $N = 31$, the number of random agents $N_r = 12$, the number of agents using S_1 and S_2 are $N_{s_1} = 6$ and $N_{s_2} = 12$, respectively. The right-hand side figure of Fig. 2 shows the performance of the intelligent agent by running the minority game for 500 times. The results show that the intelligent agent can take advantages by observing and learning from other agents' behaviors. He can predict which room is going to be the minority side with accuracy of 90% after 500 runs. However, in the macro-level (see the left-hand side figure of Fig. 2), the system still looks very random and shows no patterns.

Table 3. Impact of random agents: results are obtained by using different percentage of random agents in 500 runs of the minority game

Percentage of Random Agents	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Accuracy of Intelligent Agent (AC_I)	0.96	0.96	0.92	0.88	0.82	0.75	0.66	0.65	0.53	0.45	0.44

With higher accuracy, the intelligent agent is more likely to predict right in the next round. From the above experimental results, we can conclude that intelligent agent works almost perfectly with fixed strategies although over 30% of agents are unpredictable random agents. To study the influence of the random agents, we tested different percentage of random agents ranging from 0 to 100% and the winning accuracy of the intelligent agent is shown in table 3. Seen from the table that the accuracy of intelligent agent is almost monotonically decreasing as the percentage of random agent increases. When the percentage of random agents becomes larger than 80%, the predictive accuracy for the intelligent is around 50%. It is understandable because the dominance of random agents makes the system become totally random and unpredictable. When the accuracy is less than 50%, it means that the learning method is ineffective because even the random choice can achieve the accuracy around 50%.

2.4 Adaptive Learning

Fixed strategies for agents are not realistic assumptions in the real-world. Agents should have the ability to change their strategies adaptively based on their own records. For example, based on strategy S_1 that given $h(t) = [0011]$ the agent should take action of $S_1(h(t)) = B$ (see table 2). However, if it fails for most of time in experiments, the agent may need to adjust its strategy to choose A at the next time when see the same $h(t)$. Given a particular memory sequence $h'(t)$, agent i's *losing probability* for this sequence is calculated as the following:

$$l_i(h'(t)) = \left.\frac{\#(h'(t) = h(t) \wedge r_i(t) = Loss)}{\#h'(t) = h(t)}\right|_{(h(t),a_i(t)) \in \mathcal{D}_i} \tag{10}$$

The above equation can be interpreted as the following: given a memory string (say, [0110]), it looks for all the matching strings from the training data and see how many times the corresponding strategy $S([0110])$ will fail. If it is lower than a given threshold L, we need to adjust the strategy from A to B or vice versa. Formally, for the agent i, its strategy S can be adjusted by:

$$S(h(t)) = \begin{cases} \neg S(h(t)) & if \quad l_i(h(t)) > L_i \\ S(h(t)) & \text{otherwise} \end{cases} \tag{11}$$

where the negative function gives $\neg A = B$ and $\neg B = A$. The probability threshold for each agent i is sampled from a normal distribution: $L_i \sim \mathcal{N}(0.7, 0.12)$ in the following experiments. The reason of using a normal distribution is to increase the heterogeneous behaviors of agents in order to avoid "sudden changes" at some critical points.

In order to get sufficient statistics, the changing of strategy can only be considered after 200 runs of MGs. If the losing probability for one memory string is greater than the given threshold, we will update the strategy before continuing

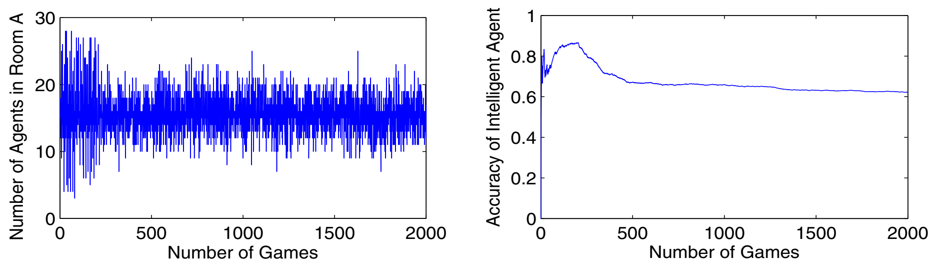

Fig. 3. Left-hand side: the number of agents in room A in 2000 runs of MG experiments. Right-hand side: the accuracy of the intelligent agent in the environment populated by a number of adaptive agents. Adaptive change of strategies starts after 200 runs.

Table 4. Impact of adaptive agents: results are obtained from 2000 runs of the minority games

Percentage of Adaptive Agents in S_1 and S_2	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
Accuracy of Intelligent Agent (AC_I)	0.86	0.78	0.73	0.65	0.59	0.58	0.55	0.51

the MG experiments. The right-hand side figure of Fig. 3 shows the accuracy of the intelligent agent with the following parameters: $N_r = 6$, $N_{S_1} = 9$ where 4 of them use adaptive strategies. $N_{S_2} = 15$ where 5 of them are adaptive agents. The intelligent agent performs very well in the first 200 runs because the strategies are fixed. After that, the learning ability has a sharp drop because of the sudden increase of the complexity. Even under these conditions that contains much uncertainties making the system so unpredictable, the accuracy of intelligent agent can still slowly stabilizing to a certain value that is around 60%. This proves the superiority of the learning model. We further test the impact of the percentage of adaptive agents and the results are shown in table 4: the accuracy of the intelligent agent AC_I is decreasing with the increasing percentage of the adaptive agents. Even in the condition that the system is populated with nearly 20% random agents (i.e. $N_r = 6$ of total $N = 31$ agents) and 70% of adaptive agents, the accuracy AC_I is still a little bit larger than 50%.

3 Learning Collective Behaviors

In previous sections, we designed an intelligent agent that uses machine learning method to learn the patterns of other agents in the MG. The experimental results show that it can predict the minority side in different scenarios. However, it assumes that the intelligent agent knows the complete information about other agents, i.e., who went to which room in which round of the game. That is not a realistic assumption. In this section, we aim to let the intelligent learn from the macro-level data $w(t)$ without knowing the records of each agent $r_i(t)$.

In this research, we still assume that N agents are divided into a few groups. One group of agents is random agents, and several groups of agents have fixed strategies of their own. However, we have no idea how many agents in each group and what strategies this group of agents employ. We only know the history of winning outcomes $w(t)$ and a guessed maximum number of groups K. In this section, we use a vector of parameters to represent the number of agents in each group and the strategy they use, then use a Genetic Algorithm [6] to optimize these parameters in order to obtain the most similar history of winning sequence. For such a problem with a large parameter space, using the Genetic Algorithm is always a good way for finding the most suitable parameters for its simpleness and effectiveness.

3.1 Fitness Function of MG

Genetic algorithms, developed by Holland [6], comprise a class of search, adaptation, and optimization techniques based on the principles of natural evolution.

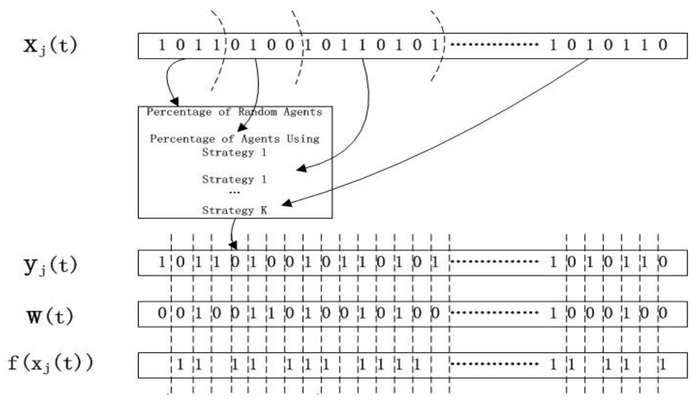

Fig. 4. The process for calculating the fitness function for a chromosome at time t. A chromosome is consisted by numbers of agents in each group and the strategies this group of agents employ. For each chromosome $\mathbf{x}_j$, we can obtain a sequence of winning outcomes $y_j(t)$ by running MGs based on the given parameters. The fitness function is calculated based on the comparisons between $y_j(t)$ and the actual sequence of winning outcomes $w(t)$.

Possible solutions are coded as *chromosomes* that evolves through generations based on a *fitness function* which evaluates the quality of the solution. Chromosomes with higher fitness are more likely to be selected to be used to reproduce off-springs. New generations of chromosomes are reproduced by *crossover* and *mutation* [10]. Successive generations are created in the same way until ultimately the final population satisfies certain criterion or reach a specified number of runs.

In our model, intelligent agent only use the information of winning outcomes $w(t)$ and a guessed maximum number of groups using fixed strategies K, such that the agents can be divided into $K+1$ groups: $\{G_r, G_1, \ldots, G_K\}$, where group G_r is the group of random agents and G_k for $k = 1, \ldots, K$ employs the strategy S_k. We use the following parameters to define one MG: percentage of random agents P_r, corresponding percentage of agents with one certain fixed strategy P_{S_k}. Therefore, each chromosome x is composed by these parameters:

$$\mathbf{x} = \{P_r, P_{S_1}, S_1, \ldots, P_{S_K}, S_K\}$$

The calculation of the fitness function $f(\mathbf{x})$ is illustrated in figure 4. At time t of the game, in order to evaluate one chromosome $\mathbf{x}_j$ ($j = 1, \ldots, J$ where J is the population size in the GA), we run the MG with the parameter setting given by $\mathbf{x}_j$ to obtain the history of winning outcomes $y_j(t)$. Comparing $y(t)$ with the actual sequence $w(t)$: for t runs from 1 to a specified time T, once $y_j(t) = w(t)$, we add one point to $f(\mathbf{x}_j)$, formally:

$$f(\mathbf{x}_j(t)) \leftarrow \begin{cases} f(\mathbf{x}_j(t)) + 1 & \text{if: } y_j(t) = w(t) \\ f(\mathbf{x}_j(t)) & \text{otherwise} \end{cases} \tag{12}$$

At each time t, the best chromosome $\mathbf{x}_{j'}$ is selected from the pool:

$$\mathbf{x}^*(t) = \arg\max_j f(\mathbf{x}_j(t)) \quad for \quad j = 1, \ldots, J$$

Given the best chromosome $\mathbf{x}^*(t)$, its parameters can give the best possible complete information of a MG so that the intelligent agent can learn with decision trees discussed in section 2.

3.2 Evolutionary Experiments

In the following experiments: we define $K = 4$, strategies S_k are generated with memory length $m = 3$. Other parameters for the MG are: $N = 81$, $P_r = 0.04$, $P_{S_1} = 0.27$, $P_{S_2} = 0.31$, $P_{S_3} = 0.16$, $P_{S_4} = 0.21$. For the genetic algorithm: crossover rate $P_c = 0.7$, mutation rate $P_m = 0.01$ and population size $J = 50$.

The upper figure of Fig. 5 shows the time-series data of how many agents went to room A in 1000 runs. This macro-data has the empirical mean nearly 41 and with a small variance. This property makes the prediction even harder because a small change of agent's choice may lead to a different winning outcome. The overall performance looks random and unpredictable.

By using the method discussed in the previous section, we use a genetic algorithm with the above parameters setting to run 1000 generations and the results are shown in the lower figure of Fig. 5. As we can see from the figure, by the increase of generations, the accuracy of the intelligent agent increase monotonically before reaching a stabilizing stage which is around 0.78.

The significance of these results is that, the collective behaviors of MGs can be decomposed into several micro-behaviors controlled by different group of agents. The behaviors of each group may follow certain simple strategies.

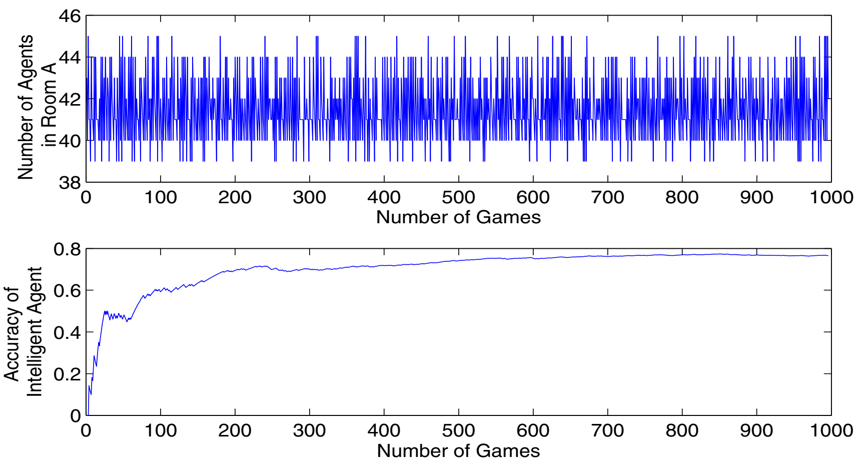

Fig. 5. Upper: Macro-level data: the number of agents in room A. Below: Accuracy of the intelligent agent by using genetic algorithm.

Combining these simple behaviors may generate complex and seemingly unpredictable macro-behaviors. Given a sequence of history wining outcomes, by using genetic algorithm, we can find the most likely combinations of single behaviors that could generate this sequence. Many real-world complex phenomena are related to the minority game [7,8,4] such as the fluctuations of stocks and currency exchange rates, although the macro-level data seemingly are random and unpredictable. Researchers could use the history of macro-level data and build models with this MG+GA framework to reconstruct the mechanism for generating these data in order to predict possible future results. This research points a new way of using the minority game model and evolutionary optimization in understanding the relationship between micro-data and macro-data. Our future work is to use real-world data to study some complex real-world phenomena that could be decomposed and explained by using this MG+GA framework.

4 Conclusions

In this paper, we proposed a method of designing intelligent agents which can learn from other agents' patterns in the minority game. Two learning scenarios are considered and tested with experiments. In the environment of complete information, a simple decision tree algorithm is used for making decisions by the intelligent agent. Experiments are conducted with fixed strategies and adaptive strategies. The results show that the intelligent agent can capture the trend and make wise decisions. However, if all agents adaptively adjust their strategies, the system becomes unpredictable again.

In the environment of incomplete information, only the history of winning outcomes are known. We assume that the agents can be decomposed into a few groups of agents that may follow different strategies. We hope to find out a combination of group behaviors that most likely to generate the given collective behaviors. A genetic algorithm can be used to find such solutions and experimental results show that the intelligent agent can predict the minority side with a probability much larger than 0.5. Our future work aims to use the MG+GA framework proposed in this research to study some real-world complex adaptive systems.

Acknowledgment

This work is partially funded by NCET of Chinese Ministry of Education and Aerospace 100 Project of Beihang University.

References

1. Arthur, W.B.: Bounded rationality and inductive behavior (the El Farol problem). American Economic Review 84, 406 (1994)
2. Challet, D., Marsili, M., Zhang, Y.C.: Stylized facts of financial markets and market crashes in minority games. Physica A 294, 514 (2001)

3. Challet, D., Marsili, M., Zecchina, R.: Statistical mechanics of systems with heterogeneous agents: Minority Games. Phys. Rev. Lett. 84, 1824 (2000)
4. Challet, D., Zhang, Y.C.: Emergence of cooperation in an evolutionary game. Physica A 246, 407 (1997)
5. Gode, D.K., Sunder, S.: Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. Journal of Political Economy 101(1), 119–137 (1993)
6. Holland, J.H.: Emergence: From Chaos to Order (1998)
7. Johnson, N.F., Jefferies, P., Hui, P.M.: Financial Market Complexity. Oxford University Press, Oxford (2003)
8. Lo, T.S., Hui, P.M., Johnson, N.F.: Theory of the evolutionary minority game. Phys. Rev. E 62, 4393 (2000)
9. Mantegna, R.N., Stanley, H.E.: An Introduction to Econophysics: Correlations and Complexity in Finance. Cambridge University Press, Cambridge (1999)
10. Qin, Z.: Market mechanism designs with heterogeneous trading agents. In: Proceedings of Fifth International Conference on Machine Learning and Applications (ICMLA 2006), Orlando, Florida, USA, pp. 69–74 (2006)
11. Qin, Z.: Naive Bayes classification given probability estimation trees. In: The Proceedings of ICMLA 2006, pp. 34–39 (2006)
12. Qin, Z., Lawry, J.: Decision tree learning with fuzzy labels. Information Sciences 172/1-2, 91–129 (2005)
13. Rapoport, A., Chammah, A.M., Orwant, C.J.: Prsoner's Dilemma: A Study in Conflict and Cooperation. Uni. of Michigan Press, Ann Arbor (1965)
14. Smith, V.L.: An experimental study of competitive market behavior. Journal of Political Economy 70, 111–137 (1962)
15. `http://en.wikipedia.org/wiki/El_Farol_Bar_problem`