Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-28 (2015) 520–525

Learning to Win in Evolutionary Two-person Boolean Game with Fixed Strategy Updating Rule ⋆

Dong WANG * **Hongbin MA** * **

* *Beijing Institute of Technology, Beijing, China(e-mail: wdsky2010@126.com)*
** *State Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing Institution of Technology (e-mail: mathmhb@bit.edu.cn)*

Abstract: This paper introduces an algorithm to learn the strategy updating rule for a two-person Boolean game using the records of the history strategies and game results. The two-person game in this paper is introduced as a zero-sum game along with a Boolean strategy set, and the strategies are governed by fixed Boolean functions whose arguments are the history strategies and game results with additive binary noise, which can be modeled as a stochastic Boolean dynamic system. However, for this easy-to-play game, there is no effective convenient methods to win more often. To achieve this goal, a learning algorithm based on Boolean regression and maximum-likelihood estimation is put forward to learn the strategy updating rule and the noise property using the records of the history strategies and game results. In addition, extensive simulations via actual examples have illustrated the effectiveness of the proposed learning algorithm.

Keywords: Learning, Boolean Game, System Identification, Parameter Estimation.

1. INTRODUCTION

In this paper an evolutionary Boolean game is introduced as a two-person zero-sum game along with a Boolean strategy set, and the strategies of the two players are evolutions of the history strategies with random items. In the game, we assume that the strategies of both players are under fixed logic strategy updating rules, which means each of the two players can arbitrarily combine the history strategies using logic rules with quite a few randomness, and we do not know the strategy updating rules *a priori* although they are fixed. We argue that this kind of Boolean games can be modeled as a Stochastic Boolean Dynamic Systems (SBDS) (see Ma et al. (2014)). Our main concerns are thus whether one player could win more often in the game with learning, and if so, then how could him.

Both two-person zero-sum games and classical propositional logic have been studied extensively. As such, the merits of this paper should be sought mainly in the light it sheds on how logic, algebra, dynamic game-theory and learning are intertwined. However, before expanding our deeply discussions, we shall introduce some related theory, and consequently locate our study in a big picture of related areas.

The most import related area is the traditional game theory. The traditional game theory, invariably called game theory shortly, is the study of strategic decision making. Specifically, it is the study of mathematical models of conflict and cooperation between intelligent rational decision-makers (see Myerson (1991)). From the beginning of game theory, it is mainly used in economics, political science, and psychology, as well as logic, computer science, and biology. The subject first addressed zero-sum games that one person's gains exactly equal the losses of the other participant or participants (see Neumann (1959), David (2007)). Today, however, game theory is widely applied into behavioral relations, and has developed into an umbrella term for the logical side of decision science, including both humans and non-humans (e.g. computers, insects/animals).

The most basic idea in game theory is that the possible outcomes of a zero-sum two-person game can be described by a payoff matrix, and each player will choose the saddle point to avoid loss in the worst case. In the games considered in (Neumann and Morgenstern (1944)), no evolutions or dynamics is involved in the game, and mainly matrix games are studied, where a series of concepts of equilibrium are investigated. Later development of game theory still focuses around various concepts of equilibrium, among which the most important one, Nash equilibrium (or Cournot-Nash equilibrium), is named after John Forbes Nash, who showed for the first time that (Nash (1950)) Nash equilibria (in mixed strategies) must exist for all finite games with any number of players and won his Nobel prize mainly for his contributions in game theory.

Inspired by traditional game theory, evolutionary game theory (EGT) was conceived by John Maynard Smith and George R. Price in 1973, when they tried to analyze Darwinian competition with game theory (see Maynard-Smith and Price (1973)). Unlike classical game theory, EGT focus more on the evolutions of strategy and the strategy are now updated under strategy profile dynamics. EGT has proven itself to be invaluable in helping to explain many complex and challenging aspects of biology. It has been particularly helpful in establishing the basis of altruistic behaviors within the context of Darwinian process. Despite its origin and original purpose, evolutionary game

⋆ This work was partially supported by the National Natural Science Foundation in China (NSFC) under Grant 61473038.

10.1016/j.ifacol.2015.12.181

theory has become of increasing interest to economists, sociologists, anthropologists, and philosophers.

Besides game theory and evolutionary game theory, another important related area is stochastic Boolean dynamic system (SBDS). As previously described, we argue that the Boolean games ins this paper can be modeled as a SBDS. A Boolean dynamical system is a particular kind of sequential dynamical system with Boolean states evolved step by step. Not far in the history, many models were put forward in order to study the Boolean dynamical systems, well-known models include cellular automata and Boolean networks. Recently, Cheng and Qi (see Cheng and Qi (2007)) introduced a new method, named semi-tensor product of matrices, which converts a Boolean network into a conventional discrete-time dynamics. And this approach exerts a tremendous fascination on researchers in Boolean dynamic systems.

The present paper is a preliminary contribution towards putting forward a learning algorithm based on Boolean regression to estimate the strategy of two-person Boolean games, and some simple yet non-trivial applications. The remainder of the paper is organized as follows. In Section II we make the mathematical descriptions of evolutionary two-person Boolean games and formulate the problem under study. Section III briefly presents Boolean regression, which is the basic idea of the proposed learning algorithm. We do some simulation studies and give our main results in Section V and finally concludes this paper by giving some concluding remarks in Section VI.

2. PROBLEM FORMULATION

Evolutionary two-person Boolean game is a particular kind of game that there are only two players as the name indicates. Here we consider the players in the game are with fixed strategy updating rules. In other words, the games we consider have $\{0,\ 1\}$ as the set of players. What's more, there only exists two strategies (**ON** or **OFF**, **TRUE** or **FALSE**, etc.), denoted by 0 and 1, for both players as well, but the strategy updating rules for them are boolean functions of the history strategies and game results. Of course, there can be randomness in the strategy updating rule. For instance, two folks, named A and B, play palms back game in several times, if their hands are in the same sides, player A wins, otherwise, player B wins. For player A (or player B), whether he choose palms or back next time depend on the previous strategies of both A and B and whether he won or lost in last game. In other words, the strategy he will use is the output of his strategy updating function, and the inputs of the function are their previous strategies and game results. Of course, he may sometimes make mistakes intentionally or not, resulting in although the output of his strategy updating function is 1, he choose 0 as his strategy. And this can be considered as a random item in the strategy updating function, here we may treat this item as noise.

In order to facilitate the description , we give the following definitions.

***Definition** 2.1.* A evolutionary two-person Boolean game (ETBG) with fixed strategy updating rule can be defined as a Boolean dynamic system

$$\begin{cases} x_0(k+1) = F_0((X_0(k)), X_1(k), Y(k)) \\ x_1(k+1) = F_1((X_0(k)), X_1(k), Y(k)) \\ y(k) = H(x_0(k), x_1(k)), \end{cases} \tag{1}$$

where $k = 1, 2, 3, \cdots$ denotes the discrete time, $i \in \{0,\ 1\}$ denotes the two players, $X_i(k) = [x_i(k),\ x_i(k-1), \cdots, x_i(0)]^T$ denotes all the history strategies of Player i, $x_i(k) \in \{0,\ 1\}$ denotes the strategy of Player i at time k, $Y(k) = [y(k),\ y(k-1), \cdots, y(0)]^T$ denotes all the history winning or losing game results, $y(k) \in \{0,\ 1\}$ denotes the game result at time k, $F_i(\cdot) : \{0,\ 1\}^k \to \{0,\ 1\}$ and $H(\cdot) : \{0,\ 1\}^2 \to \{0,\ 1\}$ denote the strategy updating function of Player i and Winning-Losing (W-L) decision function respectively. Both $F(\cdot)$ and $H(\cdot)$ are Boolean functions. And we usually call $F(\cdot)$ the strategy profile dynamics and $H(\cdot)$ the Winning-Losing function.

***Definition** 2.2.* A random evolutionary two-person Boolean game (RETBG) with fixed strategy updating rule can be defined as a stochastic Boolean dynamic system

$$\begin{cases} x_0(k+1) = F_0((X_0(k)), X_1(k), Y(k)) \oplus w_0(k, p_1) \\ x_1(k+1) = F_1((X_0(k)), X_1(k), Y(k)) \oplus w_1(k, p_2) \\ y(k) = H(x_0(k), x_1(k)), \end{cases} \tag{2}$$

where $w_i(p_i) \in \{0,\ 1\},\ i = 1, 2$ denotes the random item with parameter p_i, that means the probability of $w_i = 1$ is p_i, i.e. $Pr(w_i = 1) = p_i$, and $\oplus$ denotes binary sum, so that $a \oplus b = (a + b)\ mod\ 2$.

One of the most common strategy profile dynamics is that the next strategy is only depend on the strategy and W-L result this time. We always call this kind of strategy updating function the Markov strategy profile dynamics. And in the remaining of this paper, we will mainly discuss Markov strategy profile dynamics.

Example 2.1. A random evolutionary two-person Boolean game (RETBG) with Markov strategy profile dynamics is as following

$$\begin{cases} x_0(k+1) = F_0((x_0(k)), x_1(k), y(k)) \oplus w_0(k, p_1) \\ x_1(k+1) = F_1((x_0(k)), x_1(k), y(k)) \oplus w_1(k, p_2) \\ y(k) = H(x_0(k), x_1(k)) \end{cases} \tag{3}$$

Now, imagine we are one of the two players in the game, our concern is that whether we can win more often in the game by estimating the opponent's strategy.

In order to achieve this goal, one must solve two problems as following.

***Problem** 2.1.* Is it possible that one use both the opponent's and his own history strategies to learn (identify) the strategy profile dynamics of the other player, especially when the strategy profile dynamics is with random item (noise)?

Problem (2.1) is often referred to as the so-called identifiability problem. Obviously, for the noise-free (without random item) game as (1), one can deduce the strategy updating rule if the history strategies is complete. Now we call the game is identifiable. When the game is with noise, if we could identify the property of the noise simultaneously, the game is also identifiable.

***Problem** 2.2.* Suppose that the RETBG as (2) is identifiable and we have the history strategies, then how to design an algorithm to learn the strategy profile dynamics $F(\cdot)$ and the noise parameter p?

For Problem (2.1), a host of system identification methods can be applied, or with some revises. For Problem (2.2), there are also some researches, but almost all of them are based on the optimal strategy analysis. Here we give an approach based on strategy learning to solve Problem (2.2).

3. BASIC IDEA

3.1 Boolean Regression

Whereas traditional regression analysis focus on fitting a numerical function, boolean regression commit to fit logic function on boolean data. In statistics, regression analysis is a statistical process for estimating the relationships among variables. In large number of practical situations, we desire to identify a boolean function that captures a relationship implicit in a set of observations. When a unknown function and its arguments take only two values, we observe some outputs of the function for certain values of its arguments, and we wish to infer what the unknown function is, as best as we could. The requires like this spawned boolean regression. One instance comes from building a rule base for an expert system. Suppose that we have records of a visa officer's past decisions when he processed visa applications. We would like to create a rule through these records to determine whether a visa application can be approved or not. More issues and applications of boolean regression are talked about in (E.Boros et al. (1995)).

Next, we will use a numerical example to illustrate the Boolean regression as following. Let y be a boolean variable that is dependent on a boolean vector $x = (x_1, x_2, \cdots, x_n)$. Suppose that we have observed a series of pairs of x and y such as showed in Table 1. Note that each set of inputs arose several times, and the output y were not consistent. As an example, the set of inputs denoted by $x = (0, 0, 1)$, we observed $y = 1$ fifteen times and $y = 0$ in other four. What's more, only four of the $2^3 = 8$ possible inputs sets were observed.

Table 1. An instance of observed data

Inputs			No. of observations	
x_1	x_2	x_3	$y = 0$	$y = 1$
0	0	1	4	15
0	1	0	2	13
0	1	1	5	22
1	0	1	3	19

To derive a regression model, one would assume that some boolean function $f(x)$ truly description from boolean vector x to boolean value y . But since random error originally in $f(x)$ or observation errors, there is no boolean function exactly fits the observations. Therefore, we must consider a random item in the regression model.

In traditional case, we usually written the regression model as

$$y = g(x) + \epsilon \tag{4}$$

where g is numerical function of vector x taken numerical values and ϵ is classically assumed to have a normal distribution with mean zero.

Actually, the situation will be much simpler, since there would only two possible errors, 0 and 1. We therefore write the boolean regression as follows

$$y = f(x) \oplus \varepsilon \tag{5}$$

where $\oplus$ denotes binary sum and ε can only take 0 or 1. Typically, the random error term ε are assumed to have a Bernoulli distribution, which means that is takes the value 1 with probability p and the value 0 with probability $1-p$. Of course, the distribution of ε can be assumed more complicated.

3.2 Maximum-Likelihood/Bayesian Estimation

In statistics, maximum-likelihood estimation (MLE) is one of the most crucial method to estimate the parameters of a statistical model. MLE provides the best estimates for the model's parameters in probability sense, when applied to a data set and given a statistical model. General speaking, for a fixed set of observed data and a given statistical model, MLE selects the set of values of the model parameters that maximizes the likelihood function. Intuitively, this maximizes the "agreement" of the selected model with the observed data, and for discrete random variables it indeed maximizes the probability of the observed data under the resulting distribution. MLE has wide applications in signal process and many other fiels(see Einicke et al. (2010); J.Sijbers and Dekker (2004)).

As an instance, the likelihood function $L(p)$ of Table 1 can be computed as following. Given that the model of the data in Table 1 is $y = f(x) \oplus w(p)$, where $f(\cdot) \in \mathcal{F}$ and $\mathcal{F}$ is the set of all possible $f(\cdot)$. Let $x^1 = (0,\ 0,\ 1)$, $x^2 = (0,\ 1,\ 0)$, $x^3 = (0,\ 1,\ 1)$, $x^4 = (1,\ 0,\ 1)$, and assuming that the w are independent Bernoulli random variables with unknown parameter p, then if we let $(f(x^1), f(x^2), f(x^3), f(x^4)) = (1, 1, 1, 1)$, the likelihood function will be

$$\begin{aligned} L(p) &= p^4(1-p)^{15}p^2(1-p)^{13}p^5(1-p)^{22}p^3(1-p)^{19} \\ &= p^{14}(1-p)^{69}. \end{aligned} \tag{6}$$

Next, in order to find the best suitable p for the observed data, we just need to find the p that maximizes the likelihood $L(p)$. More generally, if Y represents the set of observations actually made, the likelihood function is

$$L(p) = Pr(Y|f, p), \tag{7}$$

where $Pr(Y|f, p)$ is the conditional probability of Y given f and p. Then, we desire to compute

$$\max_{0\le p\le 1, f\in\mathcal{F}} Pr(Y|f, p). \tag{8}$$

In Bayesian approach, the noise probability p of w is supposed to have a prior distribution with a known density function π, and each $f \in \mathcal{F}$ has some known prior probability $Pr(f)$. We generally assume that the prior distributions of p and f are independent. We wish to know the posterior probability of any given pair of f and p given the observations were made. From Bayes' law, we obtain

$$Pr(f, p|Y) = \frac{Pr(Y|f, p)Pr(f)\pi(p)}{Pr(Y)} \tag{9}$$

where

$$Pr(Y) = \sum_{f\in\mathcal{F}} \int_0^1 Pr(Y|f, p)Pr(f)\pi(p)dp \tag{10}$$

and $Pr(Y|f, p)$ is given by (6).

For Bayesian approach, what we wish to do is to compute

$$\max_{0\le p\le 1, f\in\mathcal{F}} Pr(f, p|Y), \tag{11}$$

which is equivalent to compute

$$\max_{0\le p\le 1, f\in\mathcal{F}} Pr(Y|f, p)Pr(f)\pi(p). \tag{12}$$

It is clear that if the prior probabilities of f and p are both uniformly distributed, so that $Pr(f)$ and $\pi(p)$ are

constant, then the Bayesian estimation is identical to MLE.

For our strategy estimate in Problem (2.2), we don't know exactly the strategy profile dynamics or say the strategy updating rule and then the probability of the random item as priori. To solve this problem, our basic idea is to identify $F(\cdot)$ and estimate the parameter p simultaneously. That is to say, we intend to use boolean regression and Bayesian estimation (or MLE) in our strategy estimate algorithm.

4. ALGORITHM DESIGN

4.1 Strategy Updating Model

Consider a two-person Boolean game with Markov strategy profile dynamics as (3). Let Player 0 wins the game at time k if $y(k) = 0$ and otherwise Player 1 wins. Now, suppose that we are the Player 1, and we wish to win the game more often by predicting the strategies of our opponent. Obviously, we barely know the strategy profile dynamics of our opponent, who is Player 0 here. However, we know that his strategy profile dynamics is under Markov assumption, or the game rule is just like this. In boolean regression and maximum-likelihood estimation, we suppose that the data are under some boolean functions, and we derive which function is the best fit of the observations. Next, we have to establish a model that is able to cover all the possible strategy profile dynamics.

We here formulate the main part of strategy profile dynamics for Player 0 is in **OR-NOT** form as following.

$$F(\cdot) = \beta_0 \vee \beta_1 x_0 \vee \beta_2 x_1 \vee \beta_3(\neg x_0) \vee \beta_4(\neg x_1) \vee \beta_5 y \vee \beta_6(\neg y) \tag{13}$$

where $\vee$ means logic or and $\neg$ means logic not. Here, $\beta_j, j = 0, 1, \cdots, 6$ is boolean coefficients that are to be estimated, and we intuitively understand that the term appears in the function if $\beta_j = 1$ and is absent if $\beta_j = 0$. Of course, we can assume the strategy profile dynamics as any other form as long as it has a well-defined value for any given arguments.

Then we obtain the strategy profile dynamics for Player 0 is

$$x_0(k+1) = F(\cdot) \oplus w_0(k, p_1). \tag{14}$$

If we let generalized vector $x = [1, x_0, x_1, \neg x_0, \neg x_1, y, \neg y]^T$ and $\beta = [\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6]^T$, then $F(\cdot)$ can be computed as

$$F(\cdot) = SO(\beta \circ x) \tag{15}$$

where $\circ$ means Hadamard product and SO means shift or. The Hadamard product (also known as the Schur product or the entrywise product) is a binary operation that takes two matrices of the same dimensions, for two matrices, A, B, of the same dimension, $(A \circ B)_{i,j} = (A)_{i,j}(B)_{i,j}$. The shift or is a logic operation for logic vector, such that if $x = [0, 1, 0, 0]^T$, then $SO(x) = 0 \vee 1 \vee 0 \vee 0 = 1$.

For our OR-NOT form as (13), the strategy profile dynamics have $2^7 = 128$ kinds of possible forms, which are from $\beta(0) = [\beta_0, \beta_1, \cdots, \beta_6] = [0, 0, \cdots, 0]^T$ to $\beta(127) = [\beta_0, \beta_1, \cdots, \beta_6] = [1, 1, \cdots, 1]^T$. Let

$$\mathcal{B} = \begin{bmatrix} -\beta^T(0)- \\ -\beta^T(1)- \\ \vdots \\ -\beta^T(127)- \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1 \\ 0&0&0&0&0&1&0 \\ \vdots&\vdots&\vdots&\vdots&\vdots&\vdots&\vdots \\ 1&1&1&1&1&1&0 \\ 1&1&1&1&1&1&1 \end{bmatrix}, \tag{16}$$

then we can denote the forms set by $\mathcal{B}$, and thus the strategy profile dynamics set $\mathcal{F}$ can be calculated from $\mathcal{B}$.

Usually, if the strategy updating model has n parameters to be determined, the dimension of $\mathcal{B}$ will be $2^n \times n$, or say $\mathcal{B} \in R^{2^n \times n}$. And each row of $\mathcal{B}$ is associated with a kind of strategy profile dynamics.

4.2 Strategy Learning Algorithm

Once we have the strategy updating model, we can design algorithms to learn the strategy profile dynamics. In this part of our paper, we will illustrate the principle of our algorithm with an example.

Consider a game as (3), and suppose that we are Player 1. During the game, we can record the strategies and W-L (Winning-Losing) results, so that we get Table 2.

Table 2. An instance of game record

History data			# strategies Player 0 taken	
$x_0(k-1)$	$x_1(k-1)$	$y(k-1)$	$x_0(k)=0$	$x_0(k)=1$
0	0	0	31	12
0	1	1	21	43
1	0	1	42	13
1	1	0	12	25

We take the strategy updating model as OR-NOT form like (13), and we assume that the noise probability p and the prior probability of $F(\cdot)$ are both uniformly distributed. Therefore, we just need to find the maximum-likelihood estimation of $F(\cdot)$. That is to say, for all the possible $F \in \mathcal{F}$, we desire to find the one that maximize the probability of our records.

Using the method that we get the likelihood like (6) for Table 1, in Table 2, we let $x^1 = [1, 0, 0, 0]^T$, $x^2 = [1, 0, 1, 1]^T$, $x^3 = [1, 1, 0, 1]^T$ and $x^4 = [1, 1, 1, 0]^T$. Then, if we let $(F(x^1), F(x^2), F(x^3), F(x^4)) = (0, 0, 0, 0)$, the likelihood of making the records in Table 2 is the likelihood of making thirty-one correct and twelve erroneous records of $F(x^1)$, twenty-one correct and forty-three erroneous records of $F(x^2)$, and so on. Go on like this, we obtain one hundred and six correct and ninety-three erroneous records. Thus, we desire to derive

$$\max_{p, F \in \mathcal{F}} \{(1-p)^{106}p^{93}, (1-p)^{119}p^{80}, \cdots, (1-p)^{93}p^{106}\}, \tag{17}$$

where the first term corresponds to

$$(F(x^1), F(x^2), F(x^3), F(x^4)) = (0, 0, 0, 0),$$

the second to

$$(F(x^1), F(x^2), F(x^3), F(x^4)) = (0, 0, 0, 1),$$

and so on to

$$(F(x^1), F(x^2), F(x^3), F(x^4)) = (1, 1, 1, 1),$$

for a total of $2^4 = 16$ terms. And in our OR-NOT form as (13), each $F(\cdot)$ is associated with a *beta*, which is a row in $\mathcal{B}$.

Generally, if we let R represent the records set we actually made, we desire to derive

$$\max_{p,F\in\mathcal{F}} Pr(R|F,p), \tag{18}$$

where $Pr(R|F,p)$ is the likelihood function.

And from the the calculation process above, we draw that

$$Pr(R|F,p) = (1-p)^{N-e(F)}p^{e(F)}. \tag{19}$$

Here, N is the total number of records and $e(F)$ is the number of the erroneous records given F, which in Table 2 $N = 199$ and $e(F) = 93$ for $(F(x^1), F(x^2), F(x^3), F(x^4)) = (0,0,0,0)$. And F is depend on β, so we can write $e(F)$ as $e(\beta)$ and we will use this expression.

Theorem 1. The likelihood function (19) is maximized with respect to p when $p = \frac{e(\beta)}{N}$.

Proof. Take the derivation of (19) with respect to p and let it equal to 0, we have

$$\begin{aligned}\frac{d}{dp}(Pr(R|F,p)) =& -[N-e(\beta)](1-p)^{N-e(\beta)-1}p^{e(\beta)} \\ &+ e(\beta)(1-p)^{N-e(\beta)}p^{e(\beta)-1} \\ =&[e(\beta)-Np](1-p)^{N-e(\beta)-1}p^{e(\beta)-1} \\ =&0,\end{aligned} \tag{20}$$

and we obtain

$$e(\beta) - Np = 0. \tag{21}$$

Therefore, $p = \frac{e(\beta)}{N}$, and theorem 1 is proved.

Consequently, we can replace every p in (19) with $p = \frac{e(\beta)}{N}$ and maximize the resulting expression as

$$[1-\frac{e(\beta)}{N}]^{N-e(\beta)}[\frac{e(\beta)}{N}]^{e(\beta)} \tag{22}$$

over all possible β, yields all rows in $\mathcal{B}$ which is associated with all possible $F \in \mathcal{F}$. Moreover, noting that (22) takes the same value when $e(\beta)$ is replaced by $N - e(\beta)$ and its a convex function with respect to $e(\beta)$, so that (22) is largest when $e(\beta)$ takes its largest or smallest possible value, which correspond am error-maximizing and an error-minimizing estimate of F, respectively. It is obvious that in our strategy estimate problem, the error-minimizing estimate of F is what we want.

Next problem is how to compute $e(\beta)$. In the strategy updating model (13), if we let $z_0 = 1$, $z_1 = x_0$, $z_2 = x_1$, and so on to $z_6 = \neg y$, then the OR-NOT model of (13) can be shortly written as

$$F(\cdot) = \bigvee_{i=0}^{6} \beta_i z_i. \tag{23}$$

From Table 2, we can get $z(1) = [1,0,0,1,1,0,1]$, $z(2) = [1,0,1,1,0,1,0]$, $z(3) = [1,1,0,0,1,1,0]$, $z(4) = [1,1,1,0,0,0,1]$ are all the possible of z. Let $Z = \{z(1), z(2), z(3), z(4)\}$ and $K(z) = \{k|z_i = 1\}$. Denote $n_0(z)$ the number of records of z for $x_0(k) = 0$ and $n_1(z)$ the number of records of z for $x_0(k) = 1$. Using the method given in (E.Boros et al. (1995)), the error can be expressed as

$$e(\beta) = \sum_{z\in Z} n_0(z) + \sum_{z\in Z}[n_1(z) - n_0(z)] \prod_{i\in K(z)} (1-\beta_i). \tag{24}$$

Our next task is to find a β in $\mathcal{B}$ to minimize (24), which means to solve

$$\min_{\beta\in\mathcal{B}} e(\beta). \tag{25}$$

So far, if we can find such β, then we obtain the estimate of strategy profile dynamics of Player 0. With the estimate of strategy profile dynamics, we can obtain the strategy estimate $\hat{x}_0(k+1)$, so that we can design our own strategy $x_1(k+1)$ to make $y(k+1) = 1$ which represents we will win the game next time.

However, we should note that $\hat{p} = \frac{e(\hat{\beta})}{N}$ is a biased estimator of p. In (E.Boros et al. (1995)), the authors show the detailed proof.

For the instance showed in Table 2, we can calculate that $\hat{\beta} = [0,0,1,0,0,0,0]$ to minimize $e(\hat{\beta})$ to 58, and the parameter of w_0 is $\hat{p} = 58/199 = 0.2915$. That is to say, our estimate of Player 0's strategy profile dynamics is $\hat{x}_0(k+1) = x_1(k) \oplus w_0(k,\hat{p})$, where the probability of $w_0(k) = 1$ is $\hat{p} = 0.2915$.

The above calculation process results in the estimate algorithm of strategy profile dynamics for two-person Boolean game with fixed-strategy updating rule.

Algorithm 1 Learning algorithm of strategy updating rule.

Initialization: Generate the records table as Table 2, and calculate the forms set $\mathcal{B}$ as (16).
Input: The records as Table 2.
Output: The error-minimizing best fit $\hat{\beta} \in \mathcal{B}$ and the noise probability $\hat{p}$.

1: **for** $k = 1$ to K **do**
2: Using the records to get the expression of error as (24).
3: Using each row of $\mathcal{B}$ to get the value of $e(\beta)$, and find the row that minimize $e(\beta)$.
4: Calculate the estimate noise probability $\hat{p} = \frac{e(\beta)}{k}$.
5: **end for**
6: Return $\hat{\beta}$ and $\hat{p}$.

5. SIMULATION STUDY

As a brief illustration of our learning algorithm, we will give a simulation example in this section. Let us consider the palms back game. Suppose that two folks play palms back game in several times, we are one guy of them. We made an appointment in advance that we must use Markov strategy updating rule and if our hands are in the same sides, he will win the game, otherwise we win. Suppose that our opponent update his strategy like this: if he wins this time, he will use the same strategy next time, otherwise, he will change his strategy. However, his is not always like this, sometimes he will use the same strategy next time although he lost the game this time and he will change his strategy next time although he won this time with the same probability $p = 0.3$.

If we denote our opponent by 0 and ourselves 1. Therefore, the game can be written as

$$x_0(k+1) = \begin{cases} x_0(k) \oplus w_0(k, 0.3) \ , \text{ if } y(k) = 0 \\ \neg x_0(k) \oplus w_0(k, 0.3) \ , \text{ if } y(k) = 1 \end{cases},$$
$$x_1(k+1) = F_1(x_0(k), x_1(k), y(k)) \oplus w_1(k, p_1), \quad (26)$$
$$y(k) = \begin{cases} 0 \ , \text{ if } x_0(k) = x_1(k) \\ 1 \ , \text{ if } x_0(k) \neq x_1(k) \end{cases},$$

where $y(k) = 0$ means Player 0 (our opponent) wins the game at time k and $y(k) = 1$ means Player 1 (us) wins the game.

Here, $F_1(\cdot)$ is our strategy updating rule and w_1 is the random item with $Pr(w_1 = 1) = p_1$. In our simulation, we will use three kinds of strategy updating rule to illustrate that we can win more times using the estimate algorithm. The first strategy updating rule (SUR #1) is that we use random strategy, which means $F_1 = 0$ and $Pr(w_1 = 1) = 0.5$. The second strategy updating rule (SUR #2) is that we use a fixed-strategy updating rule: if we win this time, we will use the same strategy next time, otherwise, we will change his strategy. The third strategy updating rule (SUR #3) is that we firstly learn our opponent's strategy updating rule using the learning algorithm we design in last section, and then choose a strategy to win him based on the estimation.

In mathematics, our three different strategy updating rules can be expressed as following.

$$\begin{aligned} &\text{SUR } \#1\text{: } x_1(k+1) = w_1(k, 0.5) \\ &\text{SUR } \#2\text{: } x_1(k+1) = \neg(y_k \oplus x_1(k)) \\ &\text{SUR } \#3\text{: } x_1(k+1) = \neg \hat{x}_0(k+1) \end{aligned} \quad (27)$$

where $\hat{x}_0(k+1) = \hat{F}(\cdot) \oplus w_0(k, \hat{p})$ and F is our model of Player 0's strategy updating rule which we use (23) here.

Our simulation results is showed in Fig. 1. From the left column of Fig. 1, we can see that, no matter which strategy updating rule we use, the estimate of p is close to its true value while the number of games increasing, i.e., the time going on. Further more, from the right column of Fig. 1, we can draw that we can win more often in the game if we learn the strategy updating rule (SUR #3) of our opponent using the history data.

In addition, in our simulation, the best fit β that minimize the error is $\hat{\beta} = [0, 0, 1, 0, 0, 0, 0]$, which is to say the strategy updating rule of Player 0 is

$$x_0(k+1) = x_1(k) \oplus w_0(k, \hat{p}). \quad (28)$$

As a matter of fact, from (26), we can calculate that the true strategy updating rule of Player 0 is $x_0(k+1) = x_1(k) \oplus w_0(k, 0.3)$.

6. CONCLUSION

In this paper, for two-person Boolean game with fixed-strategy updating rule, we put forward an effective learning algorithm based on Boolean regression and maximum-likelihood estimation that can be used to learn the true strategy updating rule of the player.

Our efforts in verifying the proposed learning algorithm by one example indicate that our algorithm can be used to learn the true strategy updating rule of the player such that we can win more often in the game using the learning results. Based on the simulation results, as a preliminary study, we draw the conclusion that our learning algorithm

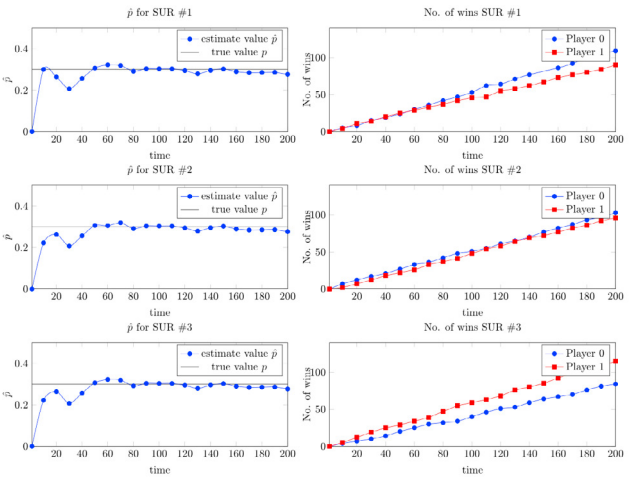

Fig. 1. Simulation results: the left column is the estimate of p, the line with solid circle dot is the estimate of p and the horizontal solid line is the true value of p. They show that the estimate of p is close to its true value while the number of games increasing, no matter which strategy updating rule we use. The right column is the No. of wins in the game of both players, the line with solid circle dot for Player 0 and the line with solid squares for Player 1. They show that Player 1 can win more often when he use strategy updating rule SUR #3 than SUR #1 and SUR #2.

can estimate the true strategy updating rule and make us win more often in the game, which need to be investigated further. Especially, rigorous mathematical proofs for the effective conditions for the algorithm.

REFERENCES

Cheng, D.Z. and Qi, H.Z. (2007). *Semi-tensor Product of Matrices Theory and Applications*. Science Press, Beijing.

David, B. (2007). The problem of waldegrave. *Journal électronique d'Histoire des Probabilités et de la Statistique*, 3(2).

E.Boros, Hammer, P., and Hooker, J. (1995). Boolean regression. *Annals of Operations Research*, 58, 201–226.

Einicke, G., Falco, G., and Malos, J. (2010). EM algorithm state matrix estimation for navigation. *IEEE Signal Processing Letters*, 17(5), 437–440.

J.Sijbers and Dekker, A. (2004). Maximum likelihood estimation of signal amplitude and noise variance from mr data. *Magnetic Resonance in Medicine*, 51(3), 586–594.

Ma, H., Wang, D., Qi, H., and Fu, M. (2014). An approach of bayesian filtering for stochastic boolean dynamic systems. *2014 26th Chinese Control and Decision Conference (CCDC)*, 4335–4340.

Maynard-Smith, J. and Price, G. (1973). The logic of animal conflict. *Nature*, 246(5427), 15.

Myerson, R. (1991). *Game Theory: Analysis of Conflict*. Harvard University Press.

Nash, J.F. (1950). Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1), 48–49.

Neumann, J.V. and Morgenstern, O. (1944). Theory of games and economic behavior. *Princeton, NJ: Princeton Univ. Press.*

Neumann, J. (1959). On the theory of games of strategy. *Contributions to the Theory of Games*, 4, 13–42.