J Syst Sci Complex (2012) 25: 209–226

TOWARDS A THEORY OF GAME-BASED NON-EQUILIBRIUM CONTROL SYSTEMS*

Yifen MU · Lei GUO

DOI: 10.1007/s11424-012-1065-6
Received: 18 March 2011 / Revised: 6 September 2011

Abstract This paper considers optimization problems for a new kind of control systems based on non-equilibrium dynamic games. To be precise, the authors consider the infinitely repeated games between a human and a machine based on the generic 2×2 game with fixed machine strategy of finite k-step memory. By introducing and analyzing the state transfer graphes (STG), it will be shown that the system state will become periodic after finite steps under the optimal strategy that maximizes the human's averaged payoff, which helps us to ease the task of finding the optimal strategy considerably. Moreover, the question whether the optimizer will win or lose is investigated and some interesting phenomena are found, e.g., for the standard Prisoner's Dilemma game, the human will not lose to the machine while optimizing her own averaged payoff when $k = 1$; however, when $k \geq 2$, she may indeed lose if she focuses on optimizing her own payoff only. The robustness of the optimal strategy and identification problem are also considered. It appears that both the framework and the results are beyond those in the classical control theory and the traditional game theory.

Key words Heterogeneous players, non-equilibrium dynamical games, optimization, state transfer graph, win-loss criterion.

1 Introduction

Over the past half century, a great deal of research effort has been devoted to adaptive control systems, and much progress has been made in both theory and applications[1−8]. A survey of some basic concepts, methods and results in adaptive systems can be found in [9], where the basic framework together with some fundamental theoretic difficulties in adaptive systems is elaborated on in detail. In the traditional parametric adaptive control, the controller may be regarded as a single agent acting on the system based on the information or measurement received, which is usually designed by incorporating an estimation mechanism: The agent (controller) makes decision based on the parameter estimates of the uncertain (time-varying) parameters that influence the dynamic behavior of the system. Note that in this framework, the time-varying parameter process may be regarded as the strategy of another 'agent', save that the action of this 'agent' usually does not depend on the control actions, and that it has no intention to gain its 'payoff'.

Yifen MU · Lei GUO
Institute of Systems Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, *China.* Email: mu@amss.ac.cn; Lguo@amss.ac.cn.
*This paper was supported by the National Natural Science Foundation of China under Grant No. 60821091 and by the Knowledge Innovation Project of Chinese Academy of Sciences under Grant No. KJCX3-SYW-S01.
◇ *This paper was recommended for publication by Editor Yiguang HONG.*

Furthermore, in many practical systems, especially social, economic, biological and ecological systems, people often encounter with the so-called complex adaptive systems (CAS) as introduced by Holland in [10]. In a CAS, as summarized in [11], a large number of components, called agents, interact with and adapt to (or learn) each other and the environment, leading to some (possibly unexpected) macro phenomena called emergence. Despite of the flexibility in modeling a wide class of complex systems by CAS, it brings great challenge to understand the evolution of a CAS mathematically, since the traditionally used mathematical tools appear to give limited help in the study of CAS, as pointed out in [11].

As an attempt towards initiating a theoretical investigation of CAS, we will, in this paper, consider a dynamic game framework that is somewhat beyond the current (adaptive) control framework. Intuitively, we will consider a scenario where two heterogeneous agents in a system, play a repeated noncooperative game[12], but the law for generating the actions of one agent is assumed to be fixed. Specifically, we will consider infinitely repeated games between a human and a machine where the machine's strategy is assumed to be fixed with k-step memory and may be unknown to the human.

We would like to point out that, to the best of the authors' knowledge, the above non-equilibrium dynamic game framework is neither contained in the traditional control theory, nor considered in the framework of the mainstream game theory. One most related direction is the differential games[12] but assumptions on the agents are actually the same as in the mainstream game theory, where all agents (players) stand in a symmetric position in either rationality or role, in order to reach some kind of equilibrium. Whereas, in our framework, the agents do not share a similar mechanism for decisions making and do not have the same level of rationality, since the two players in our framework play as the roles of a controller and a plant, respectively. And this is of fundamental importance, since in many complex systems, the agents are usually heterogenous, and they may indeed differ in either their information obtained or their ability in utilizing it. There are two points still worth mentioning.

First, it is the Stackelberg game, where the players have different roles as 'leader(s)' and 'follower(s)' and the leader can take his/her strategy first and enforce the followers to respond to it, see e.g., [12–13]. In the current paper, the follower can be regarded as the human and the leader be the machine with fixed strategy in a Prisoner's Dilemma game. In our work, the follower would like to optimize both her payoff and relative payoff over the infinite time horizon, which is different from the existing works in both the problem formulation and the analytical methods.

Second, it is worth mentioning that there have been considerable investigations in game theory in relation to adaptation and learning, which can be roughly divided into two directions. One is called evolutionary game theory[14−17], where all the agents (often in a large population) are programmed to use certain actions to play with all other agents or randomly matched, which will spread or diminish according to the payoff. The evolutionary stable equilibrium (ESS) is a key concept in the existing research. The other direction is the learning theory in games[18−19], which considers whether and how the long-run behaviors of individual agents will arrive at some equilibrium. In both the directions, the players in the games are equal in their ability to learn or adapt to the strategies of their opponents and there is no difference in roles of players and thus no hierarchy in the system. Some recent works can be found in [20–22].

This work is partly inspired by R. Axelrod's work[23−25], where in his simulation based on the Prisoner's Dilemma game, the best-played strategy emerged as a result of evolution. More researches on repeated Prisoner's Dilemma game can be found in [26–28]. The optimal strategy in this paper, however, will be obtained by optimization. To this end, we need to analyze the state transfer graph (STG) for machine strategy with k-step memory. We will show that the optimal strategy that maximizes the human's averaged payoff is actually periodic after finite

steps. General results for other games are also studied. And parts of the results in the paper were presented in [29–30], only that this paper is more complete.

The remainder of this paper is organized as follows. In Section 2, the problem is formulated. In Section 3, the state transfer graph (STG) is defined with some useful properties given. Section 4 gives the main results by answering questions proposed in Section 2. Some proofs are given in Section 5. And Section 6 concludes the paper with some remarks.

2 Problem Formulation

Consider a generic 2×2 game (there are 2 players in the game and either has 2 action options) with the payoff matrix as shown below in Figure 1.

		Player II: A	Player II: B
Player I	A	(a_{11}, b_{11})	(a_{12}, b_{12})
	B	(a_{21}, b_{21})	(a_{22}, b_{22})

Figure 1 The payoff matrix of the generic 2×2 game

Figure 1 can represent many different games. When the parameters a_{ij}, b_{ij} satisfy $a_{ij} = b_{ji}, \forall i, j$, the game is called symmetric. Below are some well known examples.

Example 2.1 The Prisoner's Dilemma (PD) game is described in Figure 2 below, where the parameters satisfy the standard conditions[23]:

$$\begin{cases} t > r > p > s, \\ r > \dfrac{t+s}{2}. \end{cases} \tag{1}$$

		Player II: C	Player II: D
Player I	C	(r, r)	(s, t)
	D	(t, s)	(p, p)

Figure 2 The payoff matrix of the Prisoner's Dilemma game

In this story, the two players are two confederate suspects who simultaneously choose their actions 'C' or 'D', where 'C' means the player cooperates with the partner, and 'D' means the player defects the partner. From (1), it is easy to see that the action profile (D,D) is the unique Nash Equilibrium, which can be dominated by (C,C) in the Pareto sense.

Example 2.2 The Snowdrift game can also be described by Figure 2, but the parameters satisfy $t > r > s > p$ and $2 \cdot r \geq t + s$. The action 'C' or 'D' means that two drivers who encounter on a snowy road, will or will not shovel the snowdrift in order to drive off the road. The profile (C,D) and (D,C) are the two pure strategy Nash equilibria here.

For all the 2×2 games described by Figure 1, the Nash equilibrium can be computed easily. The purpose of this paper is, however, not to investigate the Nash or other equilibrium. Instead, we will consider the scenario where Player I has the ability to identify her opponent and search for the best strategy in order to optimize her payoff, while Player II acts according to a given and fixed strategy. This problem is somewhat different from the traditional control problem or the classical game problem, and may be regarded as a starting point towards a theory of game-based control systems.

Vividly, let Player I be a human (we say it is a 'she' henceforth) while her opponent Player II be a machine. Assume they both know the payoff matrix. The action set of both players is denoted as $\mathbb{A} = \{C, D\}$, and the time set is discrete, $t = 0, 1, \cdots$. At time t, both players choose their actions and get their payoffs simultaneously. Let $h(t)$ denote the human's action at t and $m(t)$ the machine's.

Define the history set up to time t, H_t, as

$$H_t \triangleq \{(m(0), h(0); m(1), h(1); \cdots; m(t-1), h(t-1))\},$$

and the set of all histories is $H = \bigcup_t H_t$.

As a start, we assume the strategy of both players are deterministic mappings from H to $\mathbb{A}$. Assume that the human's action at any time t is a mapping $g(t)$ from H_t to $\mathbb{A}$. However, the machine's strategy is confined to have finite k-step-memory:

$$m(t+1) = f(m(t-k+1), h(t-k+1); \cdots; m(t-1), h(t-1); m(t), h(t)), \tag{2}$$

which, apparently, is a discrete function from $\{0,1\}^{2k}$ to $\{0,1\}$, where and hereafter, 0 and 1 stand for 'C', 'D', respectively.

Now, we define the state $s(t)$ of the game as

$$s(t) \triangleq \sum_{l=0}^{k-1} \{2^{2l+1}\ m(t-l) + 2^{2l}\ h(t-l)\} + 1. \tag{3}$$

Obviously, it establishes a one-to-one correspondence between the vector set $\{0,1\}^{2k}$ and the integer set $\{1, 2, \cdots, 2^{2k}\}$. For convenience, in what follows we denote $s(t) = s_i$ when $s(t) = i$.

When $k = 1$, the above state definition reduces to

$$s(t) = 2 \times m(t) + h(t) + 1, \tag{4}$$

which establishes a one-to-one correspondence between $(m(t), h(t))$ and the value set $s(t) \in \{s_1, s_2, s_3, s_4\}$ with $s_i = i$:

$s(t)$	$(m(t),h(t))$
s_1	(0,0)
s_2	(0,1)
s_3	(1,0)
s_4	(1,1)

Furthermore, the machine strategy (2) with $k = 1$ can be parameterized as

$$\begin{aligned} m(t+1) &= f(m(t), h(t)) \\ &= a_1 I_{\{s(t)=s_1\}} + a_2 I_{\{s(t)=s_2\}} + a_3 I_{\{s(t)=s_3\}} + a_4 I_{\{s(t)=s_4\}} \\ &= \sum_{i=1}^{4} a_i I_{\{s(t)=s_i\}}, \end{aligned} \tag{5}$$

which can be simply denoted as a vector $A = (a_1, a_2, a_3, a_4)$ with a_i being 0 or 1.

Given the strategies of both players together with the initial state, the game can be carried on and a unique sequence of states $\{s(1), s(2), \cdots\}$ will be produced, which is called a realization. Throughout this paper, $s(t)$ is assumed to be observable to both players.

Obviously, each state $s(t)$ corresponds to a unique $(m(t), h(t))$, which further corresponds to the payoffs for the human and machine, denoted by $p(t) \triangleq p(s(t))$ and $p_m(t) \triangleq p_m(s(t))$, respectively. Further, we define the relative payoff $w(t)$ of the human to the machine at t as

$$w(t) \triangleq w(s(t)) \triangleq \text{sgn}\{p(t) - p_m(t)\}, \tag{6}$$

where $\text{sgn}(\cdot)$ is the sign function and $\text{sgn}(0) = 0$. The relative payoff $w(t)$ defines whether the human win ($w(t) = 1$), or lose ($w(t) = -1$), or tie in the game at time t.

For any realizations, the averaged payoff (or ergodic payoff[31]) of the human is defined as

$$P_\infty^+ = \overline{\lim_{T\to\infty}} \ \frac{1}{T}\sum_{t=1}^{T} p(t). \tag{7}$$

When the limit exists, we simply write $P_\infty^+ = P_\infty$. Similarly, define

$$W_\infty^+ = \overline{\lim_{T\to\infty}} \ \frac{1}{T}\sum_{t=1}^{T} w(t). \tag{8}$$

The basic problems we are going to address are as follows: Q1) How can the human choose her strategy $g_t(\cdot)$ so as to obtain an optimal averaged payoff? And how is the case when the machine can make mistakes? Q2) In addition to optimality, will the human win the machine averagely, i.e., $W_\infty^+ > 0$? Q3) When the machine's strategy is unknown to the human, can she still obtain an optimal payoff?

The following sections will give some answers to these questions.

3 Analysis: The State Transfer Graph

First, note that the question Q1) raised in Section 2 is a Markov decision problem with deterministic transition probabilities, which can be solved by the algorithms in [31]. However, we care about more here like the property of the states under the solution and the answer of Q2) and Q3), about which we cannot find much help from [31]. Hence, we will solve the problem in the following new way.

Now, before defining the state transfer graph (STG) of a machine strategy, we list some basic concepts in graph theory[32]. Only finite graphs (with finite vertices and finite edges) are considered.

Let $G = (V, E)$ be a directed graph with vertex set V and edge set E.

Definition 3.1 A walk W is defined as an alternating sequence of vertices and edges, like $v_0\ e_1\ v_1\ e_2 \cdots\ v_{l-1}\ e_l v_l$, abbreviated as $v_0 v_1 \cdots v_{l-1} v_l$, where $e_i = \overline{v_{i-1} v_i}$ is the edge from v_{i-1} to v_i, $1 \le i \le l$. The total number of edges l is called the length of W.

If $v_0 = v_l$, then W is called closed, otherwise is called open.

Definition 3.2† We ignore the constraint that the length $l \ge 2$ and include 'loop' in the concept of 'cycle'. A walk W, $v_0 v_1 \cdots v_{l-1} v_l$, is called a path (directed), if the vertices $v_0, v_1, \cdots, v_l$ are distinct. A closed walk W: $v_0 v_1 \cdots v_{l-1} v_l$, $v_0 = v_l$, $l \geqslant 1$, is called a cycle (directed) if the vertices $v_1, \cdots, v_l$ are distinct .

Definition 3.3 The outdegree of a vertex v is the number of edges starting from v, denoted by $\deg^+(v)$.

Now, we are in a position to define the STG:

A directed graph with 2^{2k} vertices $\{s_1, s_2, \cdots, s_{2^{2k}}\}$ is called the state transfer graph (STG) of a given machine strategy with k-step-memory, if it contains all the possible walks representing the state transfer process of the game together with any possible human strategy. In other words, an STG contains all the one-step paths or cycles.

When $k = 1$, for a machine strategy $A = (a_1, a_2, a_3, a_4)$, the STG is a directed graph with the vertices being the state $s(t) \in \{s_1, s_2, s_3, s_4\}$ with $s_i = i$ and the edge $\overline{s_i s_j}$ exists if $s(t+1) = s_j$ can be realized from $s(t) = s_i$ by choosing $h(t+1) = 0$ or 1. Since $s_i = i$, by (4) and (5), it follows that

$$\text{the edge } \overline{s_i s_j} \text{ exists} \Leftrightarrow s_j = 2 \times a_i + 1 \text{ or } s_j = 2 \times a_i + 2, \tag{9}$$

and the way to realize this transfer is to take the human's action as $h = (s_j - 1) \bmod 2$ by (4).

By the definition, one machine strategy leads to one STG, and vice versa. The following example illustrates how to draw the STG for a given strategy.

Example 3.1 Consider the ALL C strategy $A = (0, 0, 0, 0)$ of the machine, and set the parameters in the PD game's payoff matrix as $s = 0, p = 1, r = 3, t = 5$. Then the STG of $A = (0, 0, 0, 0)$ can be drawn as Figure 3, in which $s_1(3, 0)$ means that under the state s_1, the human gets her payoff vector $P(s_1) = (p(s_1), w(s_1)) = (r, \mathrm{sgn}(r - r)) = (3, 0)$. The directed edge $\overline{s_1 s_2}$ illustrates that if the human takes action D, she can transfer the state from s_1 to s_2 with payoff vector $(5, 1)$. The rest part can be explained similarly.

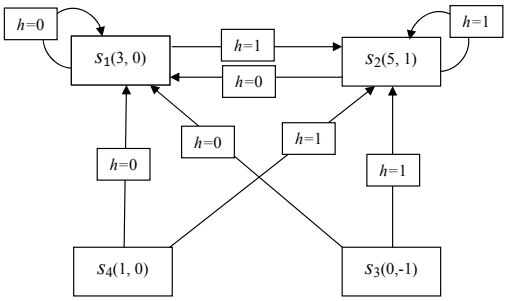

Figure 3 STG of the machine strategy ALL C $A = (0, 0, 0, 0)$

From this example, we can see that the STG contains all the useful information needed to find the optimal strategy of the human. To this end, we establish some basic properties of STG, after introducing the following definitions.

†The Definition 3.2 of cycle is a little different from [32].

Definition 3.4 A state s_j is called reachable from the state s_i, if there exists a path (or cycle) starting from s_i and ending with s_j. All the vertices which are reachable from s_i constitute a set, called the reachable set of the state s_i.

Thus, the reachability of s_j from s_i means that there exists a finite sequence of human actions, such that the state $s(\cdot)$ can be transferred from s_i to s_j.

Furthermore, we define the payoff of a walk on STG as follows.

Definition 3.5 The averaged payoff of an open walk $W = v_0 v_1 \cdots v_l$ on an STG, with $v_0 \neq v_l$, is defined as

$$p_W \triangleq \frac{p(v_0) + p(v_1) + \cdots + p(v_l)}{l+1}, \tag{10}$$

and the averaged payoff of a closed walk $W = v_0 v_1 \cdots v_l$, with $v_0 = v_l$, is defined as

$$p_W \triangleq \frac{p(v_0) + p(v_1) + \cdots + p(v_{l-1})}{l}. \tag{11}$$

Now, we give some simple properties of STG and since they are easy to prove here we will omit the proofs.

Lemma 3.1 *For a given STG, any closed walk can be divided into finite cycles, such that the edge set of the walk equals the union of the edges of these cycles. In addition, any open walk can be divided into finite cycles plus a path.*

We note here that although the partition may not be unique, it does not influence the calculation of the averaged payoff.

Lemma 3.2 *Assume that a closed walk $W = v_0 v_1 \cdots v_n$ with length L, can be partitioned into cycles $W_1, W_2, \cdots, W_m$, $m \geq 1$, with their respective lengths being $L_1, L_2, \cdots, L_m$. Then, p_W, the averaged payoff of W can be written as*

$$p_W = \sum_{j=1}^{m} \frac{L_j}{L} p_j, \tag{12}$$

where $p_1, p_2, \cdots, p_m$ are the averaged payoffs of the cycles $W_1, W_2, \cdots, W_m$, respectively.

4 Main Results

4.1 The Optimal Strategy of the Human and Its Robustness

Theorem 4.1 *For any machine strategy with finite k-step memory, there always exists a human strategy $g_t : H \to \mathbb{A}$ which is also with k-step-memory, such that the human's payoff is maximized and the resulting state sequence $\{s(t)\}_{t=1}^{\infty}$ will become periodic after some finite time.*

Remark 4.1 From the proof in Section 5.1, when the state enters into the optimal cycle, the limit P_∞ exists. Also, the optimal payoff value will depend on the initial state. This can be seen from the proof: If the two states share the same reachable set in the STG of the machine strategy, the optimal payoff will be the same, otherwise the optimal payoffs might be different.

Now, what if the machine makes mistakes with a tiny probability ε at each time? Here, we investigate the simplest case with $k = 1$. Then the machine strategy can be described as

$$\hat{m}(t) = \begin{cases} m(t), & \text{with probability } 1-\varepsilon, \\ 1 - m(t), & \text{with probability } \varepsilon, \end{cases} \tag{13}$$

where

$$m(t) = \sum_{i=1}^{4} a_i I_{\{s(t-1)=s_i\}}. \tag{14}$$

The error probability ε is assumed to be sufficiently small and known. Now, if the machine makes mistakes at $t = \tau_1, \tau_2, \cdots, \tau_i, \cdots$, it will 'jump' from one planned action to the other. Assume the human acts as if the machine does not make mistakes, the state will 'jump' as below:

$$s_1 \mapsto s_3;\ s_2 \mapsto s_4;\ s_3 \mapsto s_1;\ s_4 \mapsto\ s_2,$$

where $\mapsto$ means 'jump to'. Obviously, $\tau_i, i = 1, 2, 3, \cdots$ are random variables.

Naturally, we assume that τ_i is independent of the history of the system. Then the random sequence $\{\tau_0 = 0, \tau_1, \tau_2, \cdots, \tau_i, \cdots\}$ constitutes a homogeneous independent increment process with $\tau_i - \tau_{i-1}, i \geq 1$ satisfying a geometric distribution

$$P(\tau_i - \tau_{i-1} = l) = (1-\varepsilon)^{l-1} \cdot \varepsilon$$

with the expectation $E(\tau_i - \tau_{i-1}) = \frac{1}{\varepsilon}$. Then we have

Theorem 4.2 *For any machine strategy* (13) *and any initial state, we have*

$$P^0 \leq P^\varepsilon \leq P^0 + O(\varepsilon), \tag{15}$$

where P^0 is the expected payoff obtained using h^0, the optimal strategy of the human against the deterministic machine strategy (14)*, and P^ε is the supremum of all the expected payoffs using any human strategies.*

Remark 4.2 Theorem 4.2 implies that for any machine strategy (13) with an error probability ε, the human's optimal strategy against the deterministic machine strategy (14) can still get a near-optimal payoff, which implies its robustness in some sense.

4.2 Can the Human Win When She Optimizes?

To answer question Q2) and investigate the relationship between the optimal payoff criteria and the win-lose criteria, we have the following theorem, showing pretty complex phenomena which can not be seen in the classical adaptive control systems.

Theorem 4.3 *Let the machine strategy be with* 1*-step memory.*

i) *Consider the general* 2×2 *game with payoff matrix as Figure* 1*. Then, except the trivial case $a_{ij} < b_{ij}$, $\forall i, j$, for any machine strategy and any initial state, there always exist such payoff parameters that the human will lose, i.e., $W_\infty^+ < 0$, when she optimizes her averaged payoff.*

ii) *For the symmetric* 2×2 *game with any payoff parameters and any initial state, there exist such* 2 *and only* 2 *machine strategies $A' = (0, 1, 0, 1)$ or $A'' = (1, 0, 1, 0)$, that the the human's optimal strategy against them will never lose.*

In the Appendix 1, we will give the specific conditions of the payoff parameter as a table in Figure 5 to illustrate Theorem 4.3 ii). Considering the win-lose criterion for particular games in Examples 2.1 and 2.2, we have

Proposition 4.1 *Let the machine's strategy be with* 1*-step memory.*

i) *For the Prisoner's Dilemma game, the human's optimal strategy will not lose to any machine.*

ii) *For the Snowdrift game, there exists such machine strategy that the human will lose to it when she just optimizes her payoff.*

Clearly, it is the game structure that brings about this somewhat complex win-loss phenomenon. What interests us most is that such one-sided optimization problem (for the human)

may not always win even if the opponent has a fixed strategy. This phenomenon should be noticed for optimizers who face more than 1 criterions.

If the machine strategy has a larger memory length k, and thus it has a larger strategy set, then intuitively, it will become harder for the human to win. The proposition below shows it is true.

Proposition 4.2 *For the Prisoner's Dilemma game, when $k = 2$, there exist such machine strategies, that the human's optimal strategy will lose.*

Proposition 4.2 can be proved by a counterexample for the machine strategy '2 Tits for 1 Tat'. The details are in Section 5.

4.3 How to Find the Optimal Strategy of the Human?

By Theorem 4.1, the repeated PD game will enter a cycle of states under the optimal human strategy. This enables us to find the optimal human strategy by searching the optimal elementary cycle on the STG. We illustrate this by the example below.

Example 4.1 Let the machine strategy be ALL C $A = (0, 0, 0, 0)$, and set the parameters in the PD game's payoff matrix as $s = 0, p = 1, r = 3, t = 5$. Take the initial state as $s(0) = s_3 = (1, 0)$. The reachable set of s_3 is $\{s_1, s_2\}$, thus, it is enough to draw the induced transfer 'subgraph' as in Figure 4.

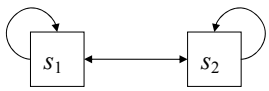

Figure 4 The transfer subgraph of ALL C strategy with $s(0) = s_3$

Obviously, there are three elementary cycles on the graph

$$W_1 = \{s_1\}, \quad W_2 = \{s_2\}, \quad W_3 = \{s_1, s_2\}$$

and by (10), the averaged payoffs of the human are, respectively,

$$p_{W_1} = p(s_1) = 3, \quad p_{W_2} = p(s_2) = 5, \quad p_{W_3} \triangleq \frac{p(s_1) + p(s_2)}{2} = 4.$$

Apparently, the optimal payoff corresponds to the cycle $W_2 = \{s_2\}$. To induce the system state enters into this cycle, the human just take $h(1) = 1$ and $h(t) = 1, t \geq 2$.

This search procedure, actually includes two steps: First, find all the elementary cycles (or circuits in some literature); and second, compute the payoffs of every elementary cycle by (10) and compare them. Apparently, the first step is a key one. Luckily, the search problem has been studied since long time ago in the literature[33]. In the Appendix, an effective search algorithm is given.

4.4 How to Identify the Unknown Machine's Strategy?

As we have shown, when the machine strategy is known, the human can find her optimal strategy. So a natural question is: What if the machine's strategy is unknown? One way is to identify the machine strategy (within finite steps) before optimizing. However, one must be very cautious to do identification. We will see this point in the next proposition.

A machine strategy parameterized by a vector A (like in (5) for the case of $k = 1$), is called identifiable if there exists a human strategy such that the vector A can be reconstructed from the corresponding realization and the initial state.

Proposition 4.3 i) *A machine strategy with k-step-memory is identifiable if and only if its corresponding* STG *is strongly connected.*

ii) *There exists such non-identifiable machine strategy, that the identification can lead to a worse payoff for the human.*

Part i) of Proposition 4.3 is quite intuitive and can be used to make confirm the existence of the unidentifiable machine strategy and help to find them. For example, when $k = 1$, the STG of the machine strategy $A = (0, 0, *, *)$ or $A = (*, *, 1, 1)$ is not strongly connected, so they are not identifiable: From any initial state, only part of the entries of $A = (a_1, a_2, a_3, a_4)$ can be identified. However, if the machine makes mistakes like in Theorem 4.2, the strategy will become identifiable: From any state s_i, there is a positive probability to reach any state s_j, i.e., the STG of the machine strategy will be strongly connected. Hence, any machine strategy can be completely identified provided long enough time.

Part ii) of Proposition 4.3 can be proved by giving an example: If the machine takes the non-identifiable strategy $A = (0, 1, 1, 1)$, then by acting with 'C' blindly, the human can get a payoff r by the PD payoff matrix at each time. However, once he tries to identify the machine's strategy, he may use the 'D' to probe it. Then the machine will be provoked and acts with 'D' forever. That will lead to a worse human payoff $p < r$ afterwards. Thus, identification here must be very cautious.

Below we give an identifying strategy for the human when the machine strategy has a $k = 1$ memory:

$$h(t+1) = \begin{cases} 0, & \text{if } a_{s(t)} \text{ is not known before time } t+1, \\ 1, & \text{otherwise.} \end{cases} \tag{16}$$

Proposition 4.4 *For any identifiable machine strategy with $k = 1$, it can be identified using the above human strategy with at most* 8 *steps from any initial state.*

5 Proofs of the Main Results

5.1 Proof of Theorem 4.1

Note that the optimization problem Q1) is actually a Markov decision problem where the transition probabilities are reduced to deterministic ones. So, Theorem 4.1 may be deduced from Theorem 9.1.8 in [31]. However, that proof needs concepts of optimal equations. Here we give an elementary and more intuitive proof using the property of the STG we defined for any machine strategy.

Proof We first consider the case where $k = 1$.

Note that under the conditions of Theorem 4.1, any cycle on the STG has a period not greater than 4. Without generality, the STG is assumed to be strongly connected. Then by searching on the STG, we can find a cycle W^* with period $d \le 4$ such that the corresponding averaged payoff p^* is maximized among all possible cycles.

Clearly, a sequence $\{s^*(t)\}_{t=1}^{\infty}$ on the STG can be constructed so that the state starting from $s(0)$ will enter into the cycle W^* within smallest possible steps, say, T_0. Obviously, the payoff of this realization equals p^* by definition. If the STG is not strongly connected, search on the reachable set of every state.

Next, we proceed to prove that the averaged payoff of any state sequence $\{s(t)\}_{t=1}^{\infty}$ will not be greater than p^*.

Let us consider any state sequence $\{s(1), s(2), \cdots, s(L)\}$ with length $L > 4$. By Lemma 3.1, the walk $s(1)s(2)\cdots s(L)$ on STG can be divided into finite cycles (plus a path with vertices

$e \leq 4$, if the walk is open). In the following, we need only to consider the little more complicated case of open walks.

Let the cycles be denoted as $W_1, W_2, \cdots, W_n$ with lengths being $L_1, L_2, \cdots, L_n$, and averaged payoffs being $p_1, p_2, \cdots, p_n$, respectively. Then, it is easy to see that $L = L_1 + \cdots + L_n + e$. Suppose the path is $v_1 v_2 \cdots v_e$. Then we have

$$\begin{aligned} p_L &= \frac{L_1}{L} p_1 + \cdots + \frac{L_n}{L} p_n + \frac{p(v_1) + \cdots + p(v_e)}{L} \\ &\triangleq \frac{L_1}{L} p_1 + \cdots + \frac{L_n}{L} p_n + \frac{A}{L}, \end{aligned} \tag{17}$$

where $0 \leq A \leq \alpha$ with α being a constant, and $L - 4 \leq L_1 + \cdots + L_n \leq L$.

Now, for the above L, suppose that there are m optimal cycles W^* contained in the sequence $\{s^*(t)\}_{t=1}^{L}$. Then we have $\{s^*(1), \cdots, s^*(L)\} = \{s^*(1), \cdots, s^*(T_0 - 1)\} \cup \{\text{m cycles } W^*\} \cup \{\text{remaining states of number f}\}$ with $T_0 - 1 < 4$ and $f < d \leq 4$. Then $L = T_0 - 1 + md + f$, and the averaged payoff of $\{s^*(t)\}_{t=1}^{L}$ is

$$\begin{aligned} p_L^* &= \frac{md}{L} p^* + \frac{p^*(s(1)) + \cdots + p^*(s(T_0 - 1)) + \sum p(\text{remaining states})}{L} \\ &\triangleq \frac{L_1 + \cdots + L_n}{L} p^* + \frac{md - (L_1 + \cdots + L_n)}{L} p^* + \frac{B}{L}, \end{aligned} \tag{18}$$

where $0 \leq B \leq \beta$ with β being a constant, and $L - 6 < md \leq L$.

From (17) and (18), we have

$$\begin{aligned} p_L - p_L^* &= \left\{ \frac{L_1}{L}(p_1 - p^*) + \cdots + \frac{L_n}{L}(p_n - p^*) \right\} - \frac{md - (L_1 + \cdots + L_n)}{L} p^* + \frac{A - B}{L} \\ &= \left\{ \frac{L_1}{L}(p_1 - p^*) + \cdots + \frac{L_n}{L}(p_n - p^*) \right\} + \frac{Z}{L} + \frac{A - B}{L}, \end{aligned} \tag{19}$$

where $| Z | \leq \gamma$ with γ being a constant.

Since all $p_i \leq p^*$, we have the first part of (19) satisfies

$$\frac{L_1}{L}(p_1 - p^*) + \cdots + \frac{L_n}{L}(p_n - p^*) \leq 0.$$

Consequently, by letting $L \to \infty$, we know that for any state sequence $\{s(t)\}_{t=1}^{\infty}$,

$$P_{\infty}^{+}(s(t)) = \overline{\lim_{l \to \infty}} \, p_L \leq \overline{\lim_{L \to \infty}} \, p_L^* = p^*. \tag{20}$$

From the definition of STG, by taking suitable $\{h^*(t)\}$, the human can induce the system state into the optimal cycle W^* within finite steps, and Theorem 2.1 holds for $k = 1$.

Next, when the machine strategy is of general finite k-step-memory, by a similar argument, the optimal state sequence $\{s^*(t)\}_{t=1}^{\infty}$ can still be shown to be a cycle after finite steps. In order to induce the system state into the optimal cycle, the human can choose optimal strategy g^* by solving the following equation

$$g^*(s^*(0), s^*(1), \cdots, s^*(t-1), s^*(t)) = g^*(s^*(t)) = (s^*(t+1) - 1) \mod 2.$$

Finally, by the definition of the state in (3), g^* is still with k-step-memory. This completes the proof. ∎

5.2 Proof of Theorem 4.2

Apparently $P^0 \le P^\varepsilon$.

Denote P^* as the optimal averaged payoff using the optimal strategy h^0, against the deterministic machine strategy (14). Easily we have $P^\varepsilon \le P^*$. So to prove the right-hand-side of (15), we just need to prove $P^* \le P^0 + O(\varepsilon)$.

Now, let the human use h^0. The machine will make mistakes at $\tau_i, i = 1, 2, \cdots$. Assume a sample path of τ_i being $\tau_1 = T_1, \tau_2 = T_2, \cdots$. Then the system states $s(t)$ are $s(\tau_0) \triangleq s(0), s(1), \cdots, s(T_1), \cdots, s(T_2), \cdots$. And the states between $s(\tau_l)$ and $s(\tau_{l+1})$ are composed of the transition states and the optimal cycles which may change after the 'jump'.

Now, we divide all the machine strategies into three classes:

Class i) The deterministic machine strategy $A = \{a_1, a_2, a_3, a_4\}$ corresponding to a strongly connected STG, i.e., $a_1 + a_2 > 0; a_3 + a_4 < 2$. There are 9 such strategies. Then the optimal cycle will be the same for any initial state and thus will be the same before and after the 'jump'.

For any finite N, assume that there are K 'jumps' at $\tau_1, \cdots, \tau_K$. Define $\tau_0 = 0$ and $\tau_{K+1} = N$. Assume also that there are q_l transition states which are not at the optimal cycle. Then we have $1 \le q_l \le 4$, $l = 0, 1, \cdots, K+1$, and

$$
\begin{aligned}
& P^* - P_N^0 \\
&= P^* - E\left\{\frac{p(\text{transition states near } \tau_l) + p(\text{optimal cycles on the inteval } [\tau_l, \tau_{l+1}])}{N}\right\} \\
&= P^* - E\left\{\frac{P^* \cdot (N - (q_0 + q_1 + \cdots + q_{K+1})) + p\{(q_0 + \cdots + q_{K+1}) \text{ transition states}\}}{N}\right\} \\
&= E\left\{\frac{P^* \cdot ((q_0 + \cdots + q_{K+1})) - p\{(q_0 + \cdots + q_{K+1}) \text{ transition states}\}}{N}\right\} \\
&\le E\left\{\frac{P^* \cdot ((q_0 + \cdots + q_{K+1})) - p_{\min} \cdot (q_0 + \cdots + q_{K+1})}{N}\right\} \\
&= (P^* - p_{\min}) \cdot E\frac{(q_0 + \cdots + q_{K+1})}{N} \\
&\le (P^* - p_{\min}) \cdot 4 \cdot E\frac{K+2}{N} \\
&= c \cdot \left(\varepsilon + \frac{2}{N}\right),
\end{aligned}
$$

where $c = 4 \cdot (P^* - p_{\min})$ is a positive constant, and $p_{\min}$ is the minimum payoff value in the payoff matrix.

Let $N \to \infty$, we have

$$P^* - P^0 \le c \cdot \varepsilon = O(\varepsilon),$$

i.e., $P^* \le P^0 + O(\varepsilon)$.

Class ii) The machine strategy $A = (0, 0, 0, 0)$ or $A = (1, 1, 1, 1)$ or $A = (0, 0, 0, 1)$ or $A = (0, 0, 1, 0)$, corresponding to an STG not strongly connected. Then for the former two strategies, all the states share the same reachable set, while for the latter two, the different reachable set share the same optimal cycle. Hence, the optimal cycle is also the same before and after the jump. By similar arguments to case i), we get $P^* - P^0 \le c \cdot \varepsilon = O(\varepsilon)$.

Class iii) The remaining three machine strategies $A = (0, 1, 1, 1)$, $A = (1, 0, 1, 1)$ or $A = (0, 0, 1, 1)$. Then there are two different optimal cycles from different initial states. And after τ_i, the states in one reachable set will jump to the new reachable set.

Take $A = (0, 1, 1, 1)$ as an example. The reachable set of s_1 is $\{s_1, s_2, s_3, s_4\}$ with the optimal cycle being $\{s_2\}$, while the other states will reach to $\{s_3, s_4\}$ with the optimal cycle being $\{s_4\}$. However, since the jump must happen, the state can not stay in only one cycle forever. So for the human, the optimal payoff can be obtained by the alternation of the two optimal cycles and h^0 is the corresponding optimal strategy. Compute the payoff difference between P^* and P^0 near the transition time, we have $P^* - P^0 \leq c \cdot \varepsilon$.

To sum up, for any 1-step-memory machine strategy, we have $P^* - P^0 \leq O(\varepsilon)$, which implies $P^\varepsilon \leq P^0 + O(\varepsilon)$. This completes the proof. ∎

5.3 Proof of Theorem 4.3

Proof of i) There are 16 machine strategies with 1-step-memory in total and we can analyze their STGs one by one.

Take $A = (1, 0, 0, 0)$ as an example and all the others are similar.

There are 3 possible cycles on its STG, i.e., $C_1 = \{s_2\}$, $C_2 = \{s_1, s_3\}$, $C_3\{s_1, s_4\}$ with the respective payoff vector $P_1 = (b_{12}, \mathrm{sgn}(b_{11} - a_{11}))$, $P_2 = \left(\frac{b_{11}+b_{21}}{2}, \frac{\mathrm{sgn}(b_{11}-a_{11})+\mathrm{sgn}(b_{21}-a_{21})}{2}\right)$, $P_3 = \left(\frac{b_{11}+b_{22}}{2}, \frac{\mathrm{sgn}(b_{11}-a_{11})+\mathrm{sgn}(b_{22}-a_{22})}{2}\right)$. Then the optimal cycle is C_1 and it loses at the same time, i.e., $w_1 = \mathrm{sgn}(b_{11} - a_{11}) < 0$, if and only if

$$\begin{cases} b_{12} > \dfrac{b_{11} + b_{21}}{2}, \\ b_{12} > \dfrac{b_{11} + b_{22}}{2}, \\ b_{11} < a_{11}. \end{cases} \tag{21}$$

Easy to see that the above inequalities are solvable. Thus we get such parameters a_{ij}, b_{ij}, that the human who optimizes her payoff only will lose to the machine strategy $A = (1, 0, 0, 0)$. Similarly, we can get the parameters conditions under which the cycles C_2, C_3 are optimal but loses. ∎

Theorem 4.3 ii) can be obtained similarly.

5.4 Proof of Propositions 4.1 and 4.2

Here we only prove the conclusion for the PD game by contradiction argument, thanks to the relationships among the payoff parameters $t > r > p > s$ and $2 \cdot r > t + s$. The statement for the Snowdrift game is easy.

Proof for Proposition 4.1 i) We need only to prove that, when $k = 1$, for any machine strategy, on the optimal cycle, the human will not lose to the machine.

Note that by (6), $w(s_1) = w(s_4) = 0$, $w(s_2) = 1$, and $w(s_3) = -1$. Hence, in the optimal cycle with $w < 0$, there must be a s_3 and no s_2. This fact makes the optimal cycle one of the following forms: $\{s_3\}$, $\{s_3, s_1\}$, $\{s_3, s_4\}$, $\{s_3, s_1, s_4\}$ or $\{s_3, s_4, s_1\}$. Easy to see that the optimal cycle can not be $\{s_3, s_4\}$, $\{s_3, s_1, s_4\}$ or $\{s_3, s_4, s_1\}$. So we just need to prove that for any machine strategy $A = (a_1, a_2, a_3, a_4)$, $\{s_3\}$ and $\{s_3, s_1\}$ cannot be the optimal cycle. Use the contradiction argument.

1) Suppose the optimal cycle is $\{s_3\}$, then $a_3 = 1$, so s_4 is reachable from s_3. Whatever state will be reached from s_4, we will get a new cycle with positive payoff after 3 steps at most. But s_3 corresponds to the payoff s, so the payoff of the cycle $\{s_3\}$ is smaller than the new cycle. Contradiction.

2) If the optimal cycle is $\{s_3, s_1\}$, we know $a_3 = 0, a_1 = 1$. Note that the state s_2 gives the largest human payoff t, and so if $a_2 = 0$, then the cycle $\{s_3, s_2, s_1\}$ will have more payoff; and if $a_2 = 1$, then the cycle $\{s_3, s_2\}$ will have more payoff. Contradiction.

Thus, the optimal cycle will not lose to the machine. That completes the proof. ∎

To prove Proposition 4.2, we will construct an example of the machine's strategy with $k = 2$ — '2 Tits for 1 Tat', i.e., only when the machine gets the largest payoff t from the action profile (D,C) for 2 consecutive times, it will act a 'C' 1 time in return. When the human plays against such a machine strategy, she may lose if she only concentrates on optimizing her payoff, which implies that she should not be 'too greedy' if she also cares about win or loss.

Proof of Proposition 4.2 $k = 2$, by (2), the machine will choose its action at $t+1$ based on the sequence $(m(t-1), h(t-1); m(t), h(t))$. From (3), the system state at time t becomes

$$s(t) \triangleq 8 \cdot m(t-1) + 4 \cdot h(t-1) + 2 \cdot m(t) + h(t) + 1, \tag{22}$$

and the state space can be denoted as $\{s_1, s_2, s_3, \cdots, s_{16}\}$ with $s_i = i, i = 1, 2, \cdots, 16$. The payoff of the human at each state $s(t)$ can be defined as

$$q(s(t)) \triangleq p(2 \cdot m(t) + h(t) + 1) = p((s(t) - 1) \mod 4 + 1).$$

Similar to (5), the machine's strategy can be written as

$$m(t+1) = \sum_{i=1}^{16} a_i I_{\{s(t)=i\}} \tag{23}$$

and denoted as a vector $A = (a_1, a_2, \cdots, a_{16})$ for simplicity.

Similar to (9), the STG of any machine strategy of 2-memory can be easily drawn, under the rules below,

$$\begin{aligned} s_i \text{ links to } s_j \Leftrightarrow s_j &= 4 \cdot [(i-1) \mod 4] + 2 \cdot a_i + 1, \text{ when } h = 0; \\ \text{or } s_j &= 4 \cdot [(i-1) \mod 4] + 2 \cdot a_i + 2, \text{ when } h = 1. \end{aligned} \tag{24}$$

Now, if the machine uses the strategy '2 Tits for 1 Tat', it will take the action $m(t+1) = 0$ only under the history $(m(t-1) = 1, h(t-1) = 0; m(t) = 1, h(t) = 0)$, which corresponds to the state $s(t) = s_{11}$ by (22), i.e., the strategy corresponds to the vector $A' = (a'_1, a'_2, \cdots, a'_{16})$, where $a'_{11} = 0$, $a'_i = 1$, if $i \neq 11$. Let the initial state be s_1 and the state transfer (sub) graph.

Then by searching the optimal elementary cycle on the subgraph, we see that the optimal one is $\{s_{10}, s_7, s_{11}\}$ with the averaged payoff $\frac{5}{3}$. On the other hand, it is not difficult to calculate the relative payoff of the cycle, which equals $-\frac{1}{3}$, which means the human loses. ∎

5.5 Proof of Proposition 4.4

When a machine strategy is of 1 memory, from (5) we know that

$$m(t+1) = a_{s(t)}. \tag{25}$$

Since the machine strategy is unknown, we will use the human strategy to excite the system so as to identify it.

Note that, according to (16), when the human is about to choose his action $h(t+1)$, he can only use the past information up to time t. Thus the rule (16) can be written as $h(1) = 0$, and

$$h(t+1) = \begin{cases} 0, & \text{if } s(t) \notin \{s(0), \cdots, s(t-1)\}, \quad t \geq 1, \\ 1, & \text{otherwise.} \end{cases} \tag{26}$$

Now, denote $s(0) = s(t_1)$. Consider $s(t_2), s(t_3), s(t_4)$, where $s(t_m) \neq s(t), \forall\, t = 0, 1, \cdots, t_m - 1, \forall m = 2, 3, 4$. Then by (25) and (26), we have $t_2 - t_1 \leq 2$, $t_3 - t_2 \leq 2$, $t_4 - t_3 \leq 3$. We will prove them one by one.

1) To prove $t_2 - t_1 \leq 2$, we just notice that each state in the STG can reach two states by the human's action taking 0 or 1.

2) To prove $t_3 - t_2 \leq 2$, we use the contradiction argument and suppose $t_3 - t_2 > 2$. Then, by (26), we have $h(t_2+1) = 0$ and $h(t_2+2) = 1$ and $s(t_2+1), s(t_2+2)$ belong to $\{s(t_1), s(t_2)\}$.

Now, we divide the states $\{s_1, s_2, s_3, s_4\}$ into two sets $R_1 = \{s_1, s_2\}$ and $R_2 = \{s_3, s_4\}$, which will be denoted as $I = \{i_1, i_2\}, J = \{j_1, j_2\}$, $i_1 < i_2$, $j_1 < j_2$ in the following proof at random. Hence, we have the possible cases:

i) $s(t_1), s(t_2)$ belong to the same R_i, say I. Then $s(t_2+1) = i_1$ and $s(t_2+2) = i_2$. Thus, i_1 can reach i_2 directly and vice versa. This contradicts with the identifiability.

ii) $s(t_1), s(t_2)$ belong to different R_i, i.e., I, J. Then we have $s(t_2+1) = i_1$ and $s(t_2+2) = j_2$. Thus, i_1 can reach j_1, j_2 not i_1, i_2 directly, which implies that $s(t_2) = j_2$ and hence $s(t_1) = i_1$. By (26), we have $s(t_1+1) = j_1$, which contradicts with $s(t_2) = j_2$.

3) To prove $t_4 - t_3 \leq 3$, we will use contradiction argument still. Suppose $t_4 - t_3 > 3$, then by (26), $h(t_3+1) = 0$. So $s(t_3+1) = i_1$. Moreover, $h(t_3+2) = 1$, $h(t_3+3) = 1$. Now, the ordered pair $(s(t_3+2), s(t_3+3))$ varies in the four possible combinations below.

i) $(s(t_3+2), s(t_3)+3)) = (i_2, i_2)$. Then i_1 can reach i_2 directly and vice versa, which contradicts with the identifiability.

ii) $(s(t_3+1), s(t_3+2)) = (i_2, j_2)$. Then i_2 can reach j_1 directly, which contradicts with $\{s(t_1), s(t_2), s(t_3)\} = \{i_1, i_2, j_2\}$.

iii) $(s(t_3+1), s(t_3+2)) = (j_2, i_2)$, which is similar to case 2).

iv) $(s(t_3+1), s(t_3+2)) = (j_2, j_2)$. Then, both i_1 and j_2 reach j_1, j_2 directly. By (26), j_1 must appear after i_1 or j_2. Hence, $\{s(t_1), s(t_2), s(t_3)\} = \{i_1, j_1, j_2\}$. From the identifiability, j_1 reach to i_1, i_2 directly, and $s(t_3+1) = i_1$ can be reached from j_1 only. Thus, $s(t_3) = j_1$. Thus, $s(t_1+1) \neq j_1$, which implies that $s(t_1+1) = i_1$ by $h(1) = 0$. So $s(t_1) = j_1$, which contradicts with $s(t_3) = j_1$.

Now, at time $t_4 + 1$, all the entries a_i can be identified. So all the needed steps are

$$1 + t_4 = 1 + (t_4 - t_3) + (t_3 - t_2) + (t_2 - t_1) + t_1 \leq 1 + 3 + 2 + 2 + 0 = 8.$$

Then the proof is completed. ∎

6 Conclusion Remarks

In an attempt to study adaptive systems beyond the traditional framework of adaptive control, we have, in this paper, studied some optimization and identification problems in a non-equilibrium dynamic game framework where two heterogeneous agents, called 'human' and 'machine', play a repeated 2×2 game. By using the concept and properties of the state transfer graph (STG), we are able to establish some interesting theoretical results, which we may not have in the traditional control framework. For example, we have shown that: i) The optimal strategy of the game is periodic after finite steps; ii) Optimizing one's payoff solely may lose to the opponent eventually; and iii) Probing the system improperly for identification may lead to a degraded payoff of the human. It goes without saying that there may be many implications and extensions of these results.

However, it would be more challenging to establish a mathematical theory for more complex systems, where many (possibly heterogeneous) agents interact with learning and adaptation. To that end, we hope that the current paper may serve as a starting point. Also, there are many other computational methods to handle such problems, like the Rational Learning algorithms, or Generic Programming, and so on. We would like to leave them as the future work.

References

[1] K. J. Astrom and B. Wittenmark, *Adaptive Control*, 2nd ed., Addison-Wesley, Reading, MA, 1995.

[2] L. Guo and H. Chen, The Astrom-Wittenmark self-tuning regulator revised and ELS-based adapptive trakers, *IEEE Trans. on Automatic Control*, 1991, **36**: 802–812.

[3] L. Guo and L. Ljung, Performance analysis of general tracking algorithms, *IEEE Trans. on Automatic Control*, 1995, **40**: 1388–1402.

[4] L. Guo, Self-convergence of weighted least-squares with applications to stochastic adaptive control, *IEEE Trans. on Automatic Control*, 1996, **41**: 79–89.

[5] T. L. Duncan, L. Guo, and B. Pasik-Duncan, Continous-time linear-quadratic Gaussian adaptive control, *IEEE Trans. on Automatic Control*, 1999, **44**: 1653–1662.

[6] G. C. Goodwin and K. S. Sin, *Adaptive Filtering, Prediction and Control*, Prentice-Hall, Englewood Cliffs NJ, 1984.

[7] P. R. Kumar and P. Varaiya, *Stochastic Systems: Estimation, Identification and Adaptive Control*, Prentice Hall, Englewood Cliffs NJ, 1986.

[8] M. Kristic, I. Kanellakopoulos, and P. Kokotoric, *Nonlinear Adaptive Control Design*, A Wiley-Interscience Publication, John Wiley & Sons, Inc., Canada, 1995.

[9] L. Guo, Adaptive systems theory: some basic concepts, methods and results, *Journal of Systems Science & Complexity*, 2003, **16**(2): 293–306.

[10] J. Holland, *Hidden Order: How Adaptation Builds Complexity*, Addison-Wesley, Reading, MA: 1995.

[11] J. Holland, Studying complex adaptive systems, *Journal of System Science & Complexity*, 2006, **19**(1): 1–8.

[12] T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory*, the Society for Industrial Applied Mathematics, Academic Press, New York, 1999.

[13] P. Y. Nie, L. Chen, and M. Fukushima, Dynamic programming approach to discrete time dynamic feedback Stackelberg games with independent and dependent followers, *European Journal of Operational Research*, 2006, **169**: 310–328.

[14] J. M. Smith, *Evolution and the Theory of Games*, Cambridge University Press, Cambridge, New York, 1982.

[15] J. W. Weibull. *Evolutionary Game Theory*, MIT Press, Cambridge, MA, 1995.

[16] J. Hofbauer and K. Sigmund, Evolutionary game dynamics, *Bulletin of the American Mathematical Society*, 2003, **40**: 479–519.

[17] S. R. Bulo and I. M. Bomze, Infection ans immunization: A new class of evolutionary game dynamics, *Games and Economic Behavior*, 2011, **71**: 193–211.

[18] D. Fudenberg and D. K. Levine, *The Theory of Learning in Games*, MIT Press, Cambridge, MA, 1998.

[19] E. Kalai and E. Lehrer, Rational learning leads to Nash equilibrium, *Econometria*, 1993, **61**: 1019–1045.

[20] D. P. Foster and H. P. Young, Regret testing: A simple payoff-based procedure for learning Nash equilibrium, *Theoretical Economics*, 2006, **1**: 341–367.

[21] J. R. Marden, G. Arslan, and J. S. Shamma, Joint strategy fictitious play with inertia for potential games, *IEEE Trans on Automatic Control*, 2009, **54**: 208–220.

[22] H. P. Young, Learning by trial and error, *Games and Economic Behavior*, 2009, **65**: 626–643.

[23] R. Axelrod, *The Evolution of Cooperation*, Basic Books, New York, 1984.

[24] L. Davis, *Genetic Algorithms and Simulated Annealing*, Morgan Kaufman Publishers, Inc., Los Altos, CA, 1987.

[25] R. Axelrod, *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*, Princeton University Press, Princeton, New Jersey, 1997.

[26] A. Rubinstein, Finite automata play the repeated prisoner's dilemma, *Journal of Economic Theory*, 1986, **39**: 83–96.

[27] G. Szabó and C. Töke, Evolutionary prisoner's dilemma game on a square lattice, *Physical Review E*, 1998, **58**: 69–73.

[28] M. A. Nowak, Five rules for the evolution of cooperation, *Science*, 2006, **314**: 1560–1563.
[29] Y. Mu and L. Guo, Optimization and idntification in a non-equilibrium dynamic game, *Proceedings of Joint* 48*th IEEE CDC and* 28*th CCC*, Shanghai, 2009.
[30] X. Hu, U. Jonsson, B. Wahlberg, and B. K. Ghosh, *Three Decades of Progress in Control Sciences*, Springer, Berlin, 2010.
[31] M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Inc, New York, 1994.
[32] J. B. Jensen and G. Gutin, *Digraphs: Theory, Algorithms and Applications*, Spring-Verlag, London, 2001.
[33] D. B. Johnson, Finding all the elementary circuits of a directed graph, *SIAM J. Comp*, 1975, **4**: 77–84.

Appendix 1

The conditions of $w < 0$ from the initial state s_1 for symmetric 2×2 games are in Figure5. below. The payoff parameters in Figure 1 are taken as $a_{11} = a, a_{21} = b, a_{12} = c, a_{22} = d$. The $\surd$ means that '$W < 0$ is possible'.

	b<c			*b>c*		
machine	*W*<0	conditions	opt cycle	*W*<0	conditions	opt cycle
(0,0,0,0)	√	*b>a*	s_2	×		
(0,0,0,1)	√	*b>a,*	s_2	×		
(0,0,1,0)	√	*b>a*	s_2	×		
(0,0,1,1)	√	*b>a*	s_2	×		
(0,1,0,0)	√	*d>c,b+d>2a*	$s_2\ s_4$	×		
(0,1,0,1)	×			×		
(0,1,1,0)	√	*b+d>2c,b+d>2a*	$s_2\ s_4$	√	*c>a,2c>b+d*	s_3
(0,1,1,1)	×			√	*c>a,c>d*	s_3
(1,0,0,0)	√	*2b>a+c,2b>b+d*	s_2	√	*a+c>2b,c>d*	$s_3\ s_1$
(1,0,0,1)	√	*b>d,2b>a+c*	s_2	√	*a+c>2b,a+c>2d*	$s_3\ s_1$
(1,0,1,0)	×			×		
(1,0,1,1)	×			√	*c>d*	s_3
(1,1,0,0)	√	*b>a,d>c*	$s_2\ s_4$	√	*c>d,a>b*	$s_3\ s_1$
(1,1,0,1)	×			√	*a+c>2d,a>b*	$s_3\ s_1$
(1,1,1,0)	√	*b+d>2c,b+d>2a*	$s_2\ s_4$	√	*2c>a+d,2c>b+d*	s_3
(1,1,1,1)	×			√	*c>d*	s_3

Figure 5 The conditions that $w < 0$ is possible from s_1

Appendix 2

An algorithm for searching optimal elementary cycle on an STG.
Step 0 Take any initial state of the system and fix it.
Step 1 Draw the STG according to the machine strategy and (9);
Step 2 Solve the reachable set of the initial state and get the transfer subgraph;
Step 3 Search on the transfer subgraph for all the elementary cycles and record them;
Step 4 Compute averaged payoffs of all the cycles and get the optimal one.

In the steps above, Step 3 is the key one and we discribe it in detail now. Notice that on the STG, the outdegree of any state is 2, thus, the STG can be stored as a tree: for a state s_i,

the two reachable states from s_i are called its left son when the human action $h = 0$ and its right son when the human action $h = 1$, denoted as $ls\{s_i\}$, $rs\{s_i\}$, respectively.

For a STG (or subgraph) $G = (V, E)$ with $|V| = m$ states (vertices), namely $v_1, v_2, \cdots, v_m$, we travel on it.

Step 3.1 $i = 1$, $t = 1$; Build a stack ps of length $m + 1$, set $ps(1) = v_i, t = 2$;

Step 3.2 For $t \in [2, m + 1]$,

case 1 $ps(t) =$ Empty,

Push into $ls\{ps(t - 1)\}$ if it is not NULL, else push into $rs\{ps(t - 1)\}$.

1) If $ps(t)$ equals the blossom point i, which means we get a cycle, then output states in the stack in the order $ps(1), \cdots, ps(t)$ and set $t = t$;

2) If $ps(t)$ dos not equal i and not equal NULL, set $t = t + 1$ if$t < m + 1$ and set $t = t$ else.

3) If $ps(t)$ equals NULL, set $t = t - 1$.

case 2 $ps(t) \neq$ Empty,

If $ps(t) = rs\{ps(t - 1)\}$, flip out the top state of the stack, and set $t = t - 1$.

Else $(ps(t) = ls\{ps(t - 1)\})$, flip it out and push into $rs\{ps(t - 1)\}$.

1) If the top state of the stack $ps(t)$ is NULL, flip it out and set $t = t - 1$;

2) If the top state equals the blossom state (which means that a cycle id found), output the states of the stack in order and set $t = t$;

3) If the top state does not equal NULL and the blossom state, set $t = t$ if $t = m + 1$ and set $t = t + 1$ else.

Step 3.3 Set sons of state v_i Null and set $i = i + 1$;

If $i = m + 1$, end. Else, build a stack ps of length $m + 1 - (i - 1)$, set $ps(1) = v_i, t = 2$, and go back to Step 3.2.