Towards solving 2-TBSG efficiently

Zeyu Jia, Zaiwen Wen and Yinyu Ye

ABSTRACT

Two-player turn-based stochastic game (2-TBSG) is a two-player game model which aims to find Nash equilibriums and is widely utilized in reinforcement learning and AI. Inspired by the fact that the simplex method for solving the deterministic discounted Markov decision processes is strongly polynomial independent of the discount factor, we are trying to answer an open problem whether there is a similar algorithm for 2-TBSG. We develop a simplex strategy iteration where one player updates its strategy with a simplex step while the other player finds an optimal counterstrategy in turn, and a modified simplex strategy iteration. Both of them belong to a class of geometrically converging algorithms. We establish the strongly polynomial property of these algorithms by considering a strategy combined from the current strategy and the equilibrium strategy. Moreover, we present a method to transform general 2-TBSGs into special 2-TBSGs where each state has exactly two actions.

1. Introduction

Markov decision process (MDP) is a widely used model in machine learning and operations research [1], which establishes basic rules of reinforcement learning. While solving an MDP focuses on maximizing (minimizing) the total reward (cost) for only one player, we consider a broader class of problems, the two-player turn-based stochastic games (2-TBSG) [15], which involves two players with opposite objectives. The first player aims to maximize the total reward and the second player aims to minimize the total reward. MDP and 2-TBSG have many useful applications, see [2,3,5,8,10,12,16].

Similar to MDP, every 2-TBSG has its state set and action set, both of which are divided into two subsets for each player, respectively. Moreover, its transition probability matrix describes the transition distribution over the state set conditioned on the current action, and its reward function describes the immediate reward when taking the action.

We use a strategy (policy) to denote a mapping from the state set into the action set. In our setting, we focus on discounted 2-TBSGs, where the reward in later steps is multiplied by a discount factor. Given strategies (policies) for both players, the total reward is defined to be the sum of all discounted rewards. We solve a 2-TBSG by finding its Nash equilibrium.
strategy (equilibrium strategy for short), where the first player cannot change its own strategy to obtain a larger total reward, and the second player cannot change its own strategy to obtain a smaller total reward. MDP can be viewed as a special case of 2-TBSG, where all states belong to the first player. In such cases, the equilibrium strategy agrees with the optimal policy of MDP.

MDPs have their linear programming (LP) formulations [3]. Hence, algorithms solving LP problems can be used to solve MDPs. One of the most commonly used algorithms for MDPs is the policy iteration algorithm [8], which can be viewed as a parallel counterpart of the simplex method solving the corresponding LP. In [18], both the simplex method solving the corresponding LP and the policy iteration algorithms have been proved to find the optimal policy in \( O\left(\frac{m l}{2 - \gamma}\right) \log\left(\frac{l}{1 - \gamma}\right) \), where \( m, l, \) and \( \gamma \) are the number of actions, the number of states and the discount factor, respectively. Later in [7], the bound for the policy iteration algorithm is improved by a factor \( l \) to \( O\left(\frac{m}{2 - \gamma}\right) \log\left(\frac{l}{1 - \gamma}\right) \). In [14], this bound is improved to \( O\left(\frac{m}{1 - \gamma}\right) \log\left(\frac{l}{2 - \gamma}\right) \). When the MDP is deterministic (all transition probabilities are either 0 or 1), a strongly polynomial bound independent on the discount factor is proved in [11] for the simplex policy iteration method (each iteration changes only one action): \( O\left(m^2 l^3 \log^2 l\right) \) for uniform discounted MDPs and \( O\left(m^3 l^5 \log^2 l\right) \) for nonuniform discounted MDPs.

However, there is no simple LP formulation for 2-TBSGs. The strategy iteration algorithm [13], an analogue to the policy iteration, is a commonly used algorithm in finding the equilibrium strategy of 2-TBSGs. It is a strongly polynomial time algorithm first proved in [7] with a guarantee to find the equilibrium in \( O\left(\frac{m}{2 - \gamma}\right) \log\left(\frac{l}{1 - \gamma}\right) \) iterations if the discount factor is fixed. When the discount factor is not fixed, an exponential lower bound is given for the policy iteration in MDP [4] and for the strategy iteration in 2-TBSG [6]. It is an open problem whether there is a strongly polynomial algorithm whose complexity is independent of the discount factor for 2-TBSG.

Motivated by the strongly polynomial algorithm for solving MDPs, we present a simplex strategy iteration algorithm and a modified simplex strategy iteration algorithm for 2-TBSGs. In both algorithms, each player updates in turn, where the second player always finds the best counterstrategy in its turn. In the simplex strategy iteration algorithm, the first player updates its strategy using the simplex algorithm. In the modified simplex strategy iteration algorithm, the first player updates the action leading to the largest improvement after the second player finds the optimal counterstrategy. When the second player is trivial and the transition model is deterministic, the 2-TBSG becomes an MDP and the simplex strategy iteration algorithm can find its solution in strongly polynomial time independent of the discount factor (refer to [11]), which is a property not possessed by the strategy iteration algorithm in [7].

We also develop a proof technique to prove the strongly polynomial complexity for a class of geometrically converging algorithms. This class of algorithms includes the strategy iteration algorithm, the simplex strategy iteration algorithm, and the modified simplex strategy iteration algorithm. The complexity for the strategy iteration algorithm given in [7] can be recovered by our techniques. Our techniques use a combination of the current strategy and the equilibrium strategy. We establish a bound of ratio between the difference of value from the current strategy to the equilibrium strategy and the difference of value from the combined strategy to the equilibrium strategy. Using this bound
and the geometrically converging property, we can prove that after a certain number of iterations, one action will disappear forever, which leads to strongly polynomial convergence when the discount factor is fixed. Although we have not fully answered the open problem, our algorithms and analysis point out a possible way for conquering the difficulties.

Furthermore, 2-TBSG where each state has exactly two actions can be transformed into a linear complementary problem [9]. An MDP where each state has exactly two actions can be solved by a combinatorial interior point method [17]. In this paper, we present a way to transform a general 2-TBSG into a 2-TBSG where each state has exactly two actions. The number of states in this constructed 2-TBSG is $\tilde{O}(m + l)$ (we use $\tilde{O}$ to hide log factors of $l$ and $m$). This result enables the application of both results in [9,17] to general cases.

The rest of this paper is organized as follows. In Section 2, we present some basic concepts and lemmas of 2-TBSGs. In Section 3, we describe the simplex strategy iteration algorithm and the modified simplex strategy iteration algorithm. The proof of complexity of the class of geometrically converging algorithm is given in Section 4. The transformation from general 2-TBSGs into special 2-TBSGs is discussed in the appendix.

2. Preliminaries

In this section, we present some basic concepts of 2-TBSG. Our focus here is on discounted 2-TBSGs, defined as follows.

**Definition 2.1:** A discounted 2-TBSG (2-TBSG for short) consists of a tuple $(S, A, P, r, \gamma)$, where $S = S_1 \cup S_2, A = A_1 \cup A_2$, $S_1, S_2, A_1, A_2$ are the state set and the action set of each player, respectively. While the first player seeks to maximize the sum of total discounted rewards, the second player aims to minimize the sum of total discounted rewards. $P \in \mathbb{R}^{|A| \times |S|}$ is the transition probability matrix, where $P(a, s)$ denotes the probability of the event that the next state is $s$ conditioned on the current action $a$. $r \in \mathbb{R}^{|A|}$ is the reward vector, where $r_a$ denotes the immediate reward function received using action $a$. To be convenient, we use $m = |A|$ to denote the number of actions, and $l = |S|$ to denote the number of states.

Given a state $s \in S$ in 2-TBSG setting, we use $A_s$ to denote the set of available actions corresponding to state $s$. A deterministic strategy (strategy for short) $\pi = (\pi_1, \pi_2)$ is defined such that $\pi_1, \pi_2$ are mappings from $S_1$ to $A_1$ and from $S_2$ to $A_2$, respectively. Moreover, each state $s \in S$ matches to an action in $A_s$.

For a given strategy $\pi = (\pi_1, \pi_2)$, we define the transition probability matrix $P_\pi \in \mathbb{R}^{l \times l}$ and reward function $r_\pi \in \mathbb{R}^l$ with respect to $\pi$. The $i$th row of $P_\pi$ is chosen to be the row of action $\pi(i)$ in $P$, and the $i$th element of $r_\pi$ is chosen to be the reward of action $\pi(i)$. It is easy to observe that the matrix $P_\pi$ is a stochastic matrix. We next define the value vector and the modified reward function.

**Definition 2.2:** The value vector $v^\pi \in \mathbb{R}^l$ of a given strategy $\pi = (\pi_1, \pi_2)$ is

$$v^{\pi_1,\pi_2} = v^\pi = (I - \gamma P_\pi)^{-1} r_\pi.$$
**Definition 2.3:** The modified reward function $r^\pi \in \mathbb{R}^m$ of a given strategy $\pi$ is defined as

$$r^\pi = r - (J - \gamma P)v_\pi,$$

where $J \in \mathbb{R}^{m \times l}$ is defined as

$$J_{ji} = \begin{cases} 1 & \text{if } j \in A_i, \\ 0 & \text{otherwise.} \end{cases}$$

Furthermore, for a given 2-TBSG, the optimal counterstrategy against another player’s given strategy is defined in Definition 2.4. The equilibrium strategy is given in Definition 2.5.

**Definition 2.4:** For player 2’s strategy $\pi_2$, player 1’s strategy $\pi_1$ is the optimal counterstrategy against $\pi_2$ if and only if for any strategy $\pi'_1$ of player 1, we have

$$v_{\pi_1, \pi_2} \geq v_{\pi'_1, \pi_2}.$$

Player 2’s optimal counterstrategy can be defined similarly: $\pi_2$ is the optimal counterstrategy against $\pi_1$ if and only if for any strategy $\pi'_2$, $v_{\pi_1, \pi_2} \leq v_{\pi'_1, \pi_2}$. Here for two value vector $v, v'$, we say $v \geq v'$ ($v \leq v'$) if and only if $v(s) \geq v'(s)$ ($v(s) \leq v'(s)$) $\forall s \in S$.

**Definition 2.5:** A strategy $\pi = (\pi_1, \pi_2)$ is called an equilibrium strategy, if and only if $\pi_1$ is the optimal counterstrategy against $\pi_2$, and $\pi_2$ is the optimal counterstrategy against $\pi_1$.

To describe the property of equilibrium strategies, we present Theorems 2.6 and 2.7 given in [7,15]. Theorem 2.6 indicates the existence of an equilibrium strategy.

**Theorem 2.6:** Every 2-TBSG has at least an equilibrium strategy. If $\pi$ and $\pi'$ are two equilibrium strategies, then $v^\pi = v'^\pi$. Furthermore, for any player 1’s strategy $\pi_1$ (or player 2’s strategy $\pi_2$), there always exists a player 2’s optimal counterstrategy $\pi_2$ against $\pi_1$ (player 1’s optimal counterstrategy $\pi_1$ against $\pi_2$), and for any two optimal counterstrategy $\pi_2, \pi'_2$ ($\pi_1, \pi'_1$), we have $v^{\pi_1, \pi_2} = v^{\pi_1, \pi'_2}$ ($v^{\pi_1, \pi_2} = v^{\pi'_1, \pi_2}$).

The next theorem points out a useful depiction of the value function at the equilibrium.

**Theorem 2.7:** Let $\pi^*$ be an equilibrium strategy for 2-TBSG. If $\pi_1$ is a strategy of player 1 and $\pi_2$ is player 2’s optimal counterstrategy against $\pi_1$, then we have $v^{\pi^*} \geq v^{\pi_1, \pi_2}$. The equality holds if and only if $(\pi_1, \pi_2)$ is an equilibrium strategy.

We now define the flux vector of a given strategy $\pi$.

**Definition 2.8:** The flux $x^\pi \in \mathbb{R}^m$ of a given strategy $\pi$ is defined as

$$x^\pi = \left( I - \gamma P_{\pi} \right)^{-T} 1,$$

$$x^\pi_a = 0, \quad \forall a \in A, a \notin \pi.$$
Our next lemma presents bounds and conditions of the flux vector, and the relationship among the value function, the flux vector and reduced costs. This lemma and the following several lemmas can be found in [7]. To make the paper self-contained, we briefly give their proofs.

**Lemma 2.9:** For any strategy $\pi$, we have

1. $1^T x^\pi = l / (1 - \gamma)$;
2. for any $a \in \pi$, $1 \leq x_a \leq l / (1 - \gamma)$;
3. $1^T v^\pi = (x^\pi)^T r$;
4. $v^\pi' - v^\pi = (I - \gamma P_{\pi'})^{-1} (r^\pi)_{\pi'}$, and moreover, $1^T (v^\pi' - v^\pi) = (x^\pi')^T r^\pi$.

**Proof:** Item (1) is proved by

$$1^T x^\pi = 1^T (I - \gamma P_{\pi})^{-1} 1 = [(I - \gamma P_{\pi})^{-1} 1]^T 1 = \frac{1}{1 - \gamma} 1^T 1 = \frac{l}{1 - \gamma}. $$

Item (2) is due to

$$(x^\pi)_a - 1 = \gamma P_{\pi} (I - \gamma P_{\pi})^{-1} 1 \geq 0.$$ 

This indicates that $(x^\pi)_a \geq 1, \forall a \in \pi$. Hence, we have $x^\pi \geq 0$ and $(x^\pi)_a \leq l / (1 - \gamma)$ from item (1). Finally, the last two items are obtained from

$$1^T v^\pi = 1^T (I - \gamma P_{\pi})^{-1} r_{\pi} = r_{\pi}^T (I - \gamma P_{\pi})^{-1} 1 = r_{\pi}^T (x^\pi)_a = (x^\pi)^T r,$$

and

$$v^\pi' - v^\pi = (I - \gamma P_{\pi'})^{-1} r_{\pi'} - (I - \gamma P_{\pi'})^{-1} (I - \gamma P_{\pi'}) v^\pi = (I - \gamma P_{\pi'})^{-1} [r - (I - \gamma P) v_{\pi'}]_{\pi'} = (I - \gamma P_{\pi'})^{-1} (r^\pi)_{\pi'};$$

$$1^T (v^\pi' - v^\pi) = 1^T (I - \gamma P_{\pi'})^{-1} (r^\pi)_{\pi'} = (x^\pi')^T (r^\pi)_{\pi'} = (x^\pi')^T r^\pi.$$ 

In the following, we present a lemma indicating the positiveness or negativeness of the reduced costs of optimal counterstrategies and equilibrium strategies.

**Lemma 2.10:** (1) A strategy $\pi_1$ for player 1 is an optimal counterstrategy against player 2’s strategy $\pi_2$ if only if $(r^{\pi_1,\pi_2})_{A_1} \leq 0$.

(2) A strategy $\pi_2$ for player 2 is an optimal counterstrategy against player 1’s strategy $\pi_1$ if only if $(r^{\pi_1,\pi_2})_{A_2} \geq 0$.

(3) A strategy $\pi = (\pi_1, \pi_2)$ is an equilibrium strategy if and only if it satisfies:

$$(r^{\pi_1,\pi_2})_{A_1} \leq 0, \quad (r^{\pi_1,\pi_2})_{A_2} \geq 0.$$ 

**Proof:** If $\pi_1$ and $\pi_2$ satisfy $(r^{\pi_1,\pi_2})_{A_1} \leq 0$, then for any player 1’s strategy $\pi_1'$, we have

$$v^{\pi_1',\pi_2} - v^{\pi_1,\pi_2} = (I - \gamma P_{\pi_1',\pi_2})^{-1} (r^{\pi_1,\pi_2})_{\pi_1'} = \sum_{n=0}^{\infty} \gamma^n P_{\pi_1',\pi_2} (r^{\pi_1,\pi_2})_{\pi_1'} \leq 0,$$

where the last inequality follows from $(r^{\pi_1,\pi_2})_{\pi_1'} \leq 0$ for $\pi_1' \in A_1$ and $(r^{\pi_1,\pi_2})_{\pi_2} = 0$. 

Suppose that player 1’s strategy $\pi_1$ is the optimal counterstrategy against player 2’s strategy $\pi_2$. For any $a' \in A_1$, $s \in S_1$ and $a' \not\in \pi_1$, we let
\[
\pi'_1(s_1) = \begin{cases} 
    a' & \text{if } s_1 = s; \\
    \pi_1(s_1) & \text{else.} 
\end{cases}
\]
Then again from Lemma 2.9 (4), we have
\[
x_{a'}^{\pi_1, \pi_2} r_{a'} = \mathbf{1}^T (v^{\pi_1, \pi_2} - v^{\pi_1, \pi_2}) \leq 0,
\]
where the inequality comes from the definition of equilibrium strategies. Since $a' \in \pi'_1$, we have $x_{a'}^{\pi_1, \pi_2} \geq 1$, which indicates that $r_{a'} \leq 0$. With this estimation and $r_{a}^{\pi} = 0 \forall a \in \pi$, we have proved that $(r^{\pi})_{A_1} \leq 0$. Hence, item (1) is established, and the proof of item (2) is similar. Finally, item (3) follows from items (1) and (2) directly.

3. Geometrically converging algorithms

Inspired by the simplex method solving the LP corresponding to the MDP and the strategy iteration algorithm given in [7], we propose a simplex strategy iteration (Algorithm 1) and a modified simplex strategy iteration algorithm (Algorithm 2) for 2-TBSG.

**Algorithm 1** A simplex strategy iteration method

1. **Initialize:** $\pi_1^0$ for player 1, $\pi_2^0$ for player 2, $n \leftarrow 0$.
2. **repeat**
3. Find $s \in S_1$ and $a \in A_s$ such that $r_{a}^{\pi_1^0, \pi_2^0}$ is the largest among such $(s, a)$.
4. $\pi_1^{n+1}(s) \leftarrow a, \pi_1^{n+1}(s') \leftarrow \pi_1^n(s') \forall s' \in S_1, s' \neq s$.
5. Find the optimal counterstrategy $\pi_2^{n+1}$ against $\pi_1^{n+1}$.
6. **until** $\pi_1^n = \pi_1^{n+1}$.
7. **Output:** $\pi_1^n, \pi_2^n$.

The simplex strategy iteration algorithm can be viewed as a generalization of the strongly polynomial simplex algorithm in solving MDPs [11]. In our algorithm, both players update their strategies in turn. In each iteration, while the first player updates its strategy using the simplex method, which means only updating the action with the largest reduced cost, the second player updates its strategy according to the optimal counterstrategy. When the second player has only one possible action and the transition matrix is deterministic, the 2-TBSG reduces to a deterministic MDP. Then, the simplex strategy iteration algorithm can find an equilibrium (optimal) strategy in strongly polynomial time independent of $\gamma$, which is a property that has not been proven for the strategy iteration [7].

As for the modified simplex strategy iteration algorithm, it can be viewed as a modification of the simplex strategy iteration algorithm. In this algorithm, both players also update their strategies in turn, and the second player always finds the optimal counterstrategies in its moves. However, in each of the first player’s move, only the action is updated which
Algorithm 2 A modified simplex strategy iteration algorithm

1: Initialize: $\pi_1$ for player 1, $\pi_2$ for player 2
2: Let $\pi_1^0 \leftarrow \pi_1$, $n \leftarrow 0$
3: repeat
4: \[ \pi_1^{n+1} \leftarrow \pi_1^n, \quad \pi_2^{n+1} \leftarrow \pi_2^n \]
5: for any $s \in S$ and action $a \in A_s$ do
6: \[ \text{Let } \tilde{\pi}_1^{n+1} \text{ be the player 1's strategy where only state } s \text{ 's action changes to } a, \text{ and other states' actions keep to be the same as } \pi_1^n. \]
7: \[ \text{Let } \tilde{\pi}_2^{n+1} \text{ be the optimal counterstrategy against } \tilde{\pi}_1^{n+1}. \]
8: if $1^T \nu \pi_1^{n+1} - \nu \pi_2^{n+1} \leq 1^T \nu \tilde{\pi}_1^{n+1} - \nu \tilde{\pi}_2^{n+1}$ then
9: \[ \pi_1^{n+1} \leftarrow \tilde{\pi}_1^{n+1}, \quad \pi_2^{n+1} \leftarrow \tilde{\pi}_2^{n+1}. \]
10: end if
11: end for
12: until $1^T \pi_1^n = 1^T \pi_1^{n+1}$
13: Output: $\pi_1^n, \pi_2^n$.

leads to the biggest improvement on the value function when the second player uses the optimal counterstrategy.

It is easy to know that every iteration of the simplex strategy iteration algorithm involves a step of a simplex update and a solution to an MDP. And every iteration of the modified simplex strategy iteration algorithm involves solutions to multiple MDPs. Hence, every iteration in both of these two algorithms can be solved in strongly polynomial time when the discount factor is fixed.

Next, we present a class of geometrically converging algorithms used for proving the strongly polynomial complexity for several algorithms in the next section.

Definition 3.1: We say a strategy-update algorithm (algorithms which update strategies for both players in each iteration) is a geometrically converging algorithm with parameter $a_M$, if it updates a strategy $\pi^n = (\pi^n_1, \pi^n_2)$ to $\pi^{n+1} = (\pi^{n+1}_1, \pi^{n+1}_2)$ such that the following properties hold.

- $\pi^{n+1}_2$ is the optimal counterstrategy against $\pi^{n+1}_1$;
- $(\nu \pi^n)_{\pi_1^{n+1}} \geq 0$;
- If $1^T (\nu \pi^{n+1} - \nu \pi^n) = 0$, then $\pi^n$ is an equilibrium strategy;
- The updates of this algorithm satisfy

\[
1^T \left( \nu^{\pi^n_1, \pi^n_2} - \nu^{\pi^n_1, \pi^n_2 + M} \right) \leq \frac{(1 - \gamma)^2}{l^2} \cdot 1^T \left( \nu^{\pi^n_1, \pi^n_2} - \nu^{\pi^n_1, \pi^n_2} \right).
\]

To begin with, we exhibit a lemma indicating the geometrically converging property of the value function in the simplex strategy iteration algorithm.

Lemma 3.2: Suppose the sequence of strategy generated by the simplex strategy iteration algorithm is $\pi^1 = (\pi^1_1, \pi^1_2), \pi^2 = (\pi^2_1, \pi^2_2), \ldots, \pi^n = (\pi^n_1, \pi^n_2), \ldots$ Then, the following
inequality holds

\[ 1^T (v^{π^n} - v^{π^{n+1}}) \leq \left( 1 - \frac{1 - γ}{l} \right) 1^T (v^{π^*} - v^{π^n}). \]  (1)

**Proof:** According to Algorithm 1, we have

\[ 1^T (v^{π^{n+1}} - v^{π^n}) \geq r^{π^n}_{a_1} x^{π^{n+1}}_{a_1} \geq \frac{1 - γ}{l} \sum_{a \in A_1} r^{π^n}_a x^{π^{n+1}}_a \]

\[ \geq \frac{1 - γ}{l} \sum_{a \in A} r^{π^n}_a x^{π^{n+1}}_a = \frac{1 - γ}{l} 1^T (v^{π^*} - v^{π^n}), \]

where the second and third inequalities follow from Lemma 2.9 (2) and the choice of \( a_1 = \arg\max_{a \in A_1} r^{π^n}_a \), the fourth inequality follows from Lemma 2.10, and the first inequality and last equation are due to Lemma 2.9 (4) and Lemma 2.10.

Using this lemma, we show in the next proposition that the strategy iteration algorithm, Algorithms 1 and 2 all belong to the class of geometrically converging algorithms.

**Proposition 3.3:** (1) The strategy iteration algorithm given in [7] is a geometrically converging algorithm with parameter \( M = \mathcal{O}(\frac{1 - γ}{l}) \log(l/(1 - γ)). \)

(2) The simplex strategy iteration algorithm (Algorithm 1) is a geometrically converging algorithm with parameter \( M = \mathcal{O}((l/(1 - γ)) \log(l/(1 - γ))). \)

(3) The modified simplex strategy iteration algorithm (Algorithm 2) is a geometrically converging algorithm with parameter \( M = \mathcal{O}((l/(1 - γ)) \log(l/(1 - γ))). \)

**Proof:** It is easy to verify that the previous described three algorithms satisfy the first three conditions in the definition of geometrically converging algorithms. Next, we prove that all of these algorithms satisfy the last condition. For the strategy iteration algorithm, according to Lemmas 4.8 and 5.4 given in [7], we have

\[ 1^T (v^{π^*} - v^{π^{n+1}}) \leq γ 1^T (v^{π^*} - v^{π^n}). \]

Hence, if

\[ M = \frac{2c_1}{1 - γ} \log \frac{l}{1 - γ} = \mathcal{O} \left( \frac{1}{1 - γ} \log \frac{l}{1 - γ} \right) \quad (c_1 \geq 1 \text{ is a constant}), \]

then we obtain

\[ 1^T (v^{π^*} - v^{π^{n+M}}) \leq γ^M 1^T (v^{π^*} - v^{π^n}) \leq γ^{-2} \log_{γ^l/1} 1^T (v^{π^*} - v^{π^n}) \]

\[ = \left( \frac{1 - γ^2}{l^2} \right) 1^T (v^{π^*} - v^{π^n}), \]

and the last condition of geometrically converging algorithms is verified.
For the simplex strategy iteration algorithm, if we choose $M = O((l/(1 - \gamma)) \log (l/(1 - \gamma)))$ ($c_2 \geq 1$ is a constant), then according to inequality (1), we have

$$1^T (v^{\pi^*} - v^{\pi_{n+M}}) \leq \frac{(1 - \gamma)^2}{l^2} 1^T (v^{\pi^*} - v^{\pi_n}),$$

and the last condition of geometrically converging algorithms is verified.

Finally, we consider the modified simplex strategy iteration algorithm. For $n \geq 2$, let $a_1 = \arg \max_{a \in A_1} r_a^{\pi_n}$, where $a_1$ is an action of state $s_1$. Let

$$\pi'_1(s) = \begin{cases} a_1 & \text{if } s = s_1, \\ \pi^n(s) & \text{others,} \end{cases}$$

$\pi'_1$ be player 2’s optimal counterstrategy against $\pi'_1$, and $\pi' = (\pi'_1, \pi'_2)$. Then from inequality (1), we have

$$1^T (v^{\pi^*} - v^{\pi'}) \leq \left( 1 - \frac{1 - \gamma}{l} \right) 1^T (v^* - v^{\pi^n}).$$

According to Algorithm 2, we have

$$1^T v^{\pi_{n+1}} \geq 1^T v^{\pi'},$$

which leads to the following estimation:

$$1^T (v^{\pi^n} - v^{\pi_{n+1}}) \leq \left( 1 - \frac{1 - \gamma}{l} \right) 1^T (v^* - v^{\pi^n}).$$

Therefore, similar to the previous case we can choose $M = O((l/(1 - \gamma)) \log (l/(1 - \gamma)))$ such that

$$1^T (v^{\pi^*} - v^{\pi_{n+M}}) \leq \frac{(1 - \gamma)^2}{l^2} 1^T (v^{\pi^{\pi^*}} - v^{\pi^n}),$$

and the last condition of geometrically converging algorithms is verified.

\[\square\]

4. Strongly polynomial complexity of geometrically converging algorithms

In this section, we develop the strongly polynomial property of geometric converging algorithms if the parameter $M$ is viewed as a constant. Slightly different from the proof in [7] for the strategy $(\pi^n_1, \pi^n_2)$ at the $n$th iteration, we present a proof by considering the strategy $(\pi^*_1, \pi^*_2)$, where $(\pi^*_1, \pi^*_2)$ is an equilibrium strategy. We show that $1^T (v^{\pi^n_1, \pi^n_2} - v^{\pi^*_1, \pi^*_2})$ can be both upper and lower bounded by some proportion of $1^T (v^{\pi^n_1, \pi^n_2} - v^{\pi^*_1, \pi^*_2})$. By applying the property of geometrically converging algorithms, we obtain that after a certain number of iterations, a player 1’s action will disappear in $\pi^n_1$ forever.

**Theorem 4.1**: Any geometrically converging algorithm with a parameter $M$ finds the equilibrium strategy in

$$O(Mm)$$

number of iterations.
\textbf{Proof:} Suppose }\pi^1 = (\pi_1^1, \pi_2^1), \pi^2 = (\pi_1^2, \pi_2^2), \ldots, \pi^n = (\pi_1^n, \pi_2^n)\text{ is the sequence generated by a geometrically converging algorithm. We define } \eta^n = (\pi_1^n, \pi_2^n), \text{ where } \pi^* = (\pi_1^*, \pi_2^*)\text{ is one of the equilibrium strategy.}

According to Lemma 2.10 and the fact that }\pi_2^{n+1}\text{ is the optimal counterstrategy against }\pi_1^{n+1}, \text{ and the definition of geometrically converging algorithm, we have }

\begin{equation*}
1^T(\nu^{n+1} - \nu^n) = \sum_{a \in \pi_1^{n+1}} x_a^{n+1} r_a^{n+1} + \sum_{a \in \pi_2^{n+1}} x_a^{n+1} r_a^{n+1} \geq \sum_{a \in \pi_1^n} x_a^n r_a^n \geq 0,
\end{equation*}

which directly leads to

\begin{equation}
1^T \nu^n \leq 1^T \nu^{n+1}.
\end{equation}

According to Lemma 2.10, we have

\begin{equation*}
1^T(\nu^n - \nu^*) = 1^T(\pi_1^* \pi_2^* - \pi_1^n \pi_2^n) = -(\pi_1^n, \pi_2^n)^T r_{1,1}^* \pi_2^* = -(\pi_1^n, \pi_2^n)^T A_1 r_{1,1}^* \geq 0,
\end{equation*}

\begin{equation*}
1^T(\eta^n - \nu^n) = 1^T(\pi_1^n, \pi_2^n - \pi_1^n \pi_2^n) = (\pi_1^n, \pi_2^n)^T r_{1,1}^* \pi_2^n = (\pi_1^n, \pi_2^n)^T A_1 r_{1,1}^* \geq 0,
\end{equation*}

which implies

\begin{equation}
1^T \nu^n \leq 1^T \nu^* \leq 1^T \nu^n.
\end{equation}

We next prove the following inequality:

\begin{equation}
1^T(\nu^n - \nu^*) \geq \frac{1 - \gamma}{l} \cdot 1^T(\nu^n - \nu^*).
\end{equation}

A direct calculation gives

\begin{equation*}
1^T(\nu^n - \nu^*) = 1^T(\pi_1^n, \pi_2^n - \pi_1^n \pi_2^n) = -(\pi_1^n, \pi_2^n)^T r_{1,1}^* \pi_2^* = - \sum_{a \in \pi_1^n} x_a^n \pi_2^* r_a^* \leq - \sum_{a \in \pi_1^n} x_a^n \pi_2^* r_a^*,
\end{equation*}

where the last inequality is obtained from Lemma 2.10. Then noticing that

\begin{equation*}
1 \leq x_a^n \pi_2^*, \quad x_a^n \pi_2^* \leq \frac{l}{1 - \gamma}, \quad r_a^* \pi_2^* \leq 0, \quad \forall a \in \pi_1^n,
\end{equation*}

we have

\begin{equation*}
1^T(\nu^n - \nu^*) = 1^T(\pi_1^n, \pi_2^n - \pi_1^n \pi_2^n) = -(\pi_1^n, \pi_2^n)^T r_{1,1}^* \pi_2^* = - \sum_{a \in \pi_1^n} x_a^n \pi_2^* r_a^* \pi_2^* \geq - \frac{1 - \gamma}{l} \sum_{a \in \pi_1^n} x_a^n \pi_2^* r_a^* \pi_2^* \geq \frac{1 - \gamma}{l} 1^T(\nu^n - \nu^*).
\end{equation*}

Then, the inequality (4) is proved.
Finally, we prove that for any $n$, either there exists an action $a_1$ in $\pi^n_1$ will never belong to $\pi^n_{1+m}$ when $m > M$, or we have

$$1^T(v^{\pi^n_{1+m+1}} - v^{\pi^n_{1+m}}) = 0.$$ 

Actually for any $p > M$, suppose $1^T(v^{\pi^n_{1+m}} - v^{\pi^n_{1+m+1}}) \neq 0$, we obtain

$$1^T(v^{\pi^n} - v^{\pi^n+1}) < 1^T(v^{\pi^n} - v^{\pi^n+M}) \leq \frac{(1 - \gamma)^2}{l^2} 1^T(v^{\pi^n} - v^{\pi^n}).$$

from (2) and the definition of geometrically converging algorithm. Hence, according to (3) and (4), we get

$$1^T(v^{\pi^n} - v^{\pi^n+1}) \leq 1^T(v^{\pi^n} - v^{\pi^n+M}) \leq \frac{1 - \gamma}{l} 1^T(v^{\pi^n} - v^{\pi^n}).$$

Therefore, choosing $a_1 = \arg\min_{a \in \pi^n_1} \ r^{\pi^n}_{a} \leq 0$, and because for any $a \in \pi^n_1$, $r^{\pi^n}_{a} \leq 0$ according to Lemma 2.10, we obtain

$$1^T(v^{\pi^n} - v^{\pi^n_1}) = - \sum_{a \in \pi^n_1} x^{n}_{a} r^{\pi^n}_{a} \leq \left( \sum_{a \in \pi^n_1} x^{n}_{a} \right) \cdot (-r^{\pi^n}_{a_1}) \leq - \frac{1}{1 - \gamma} \cdot r^{\pi^n}_{a_1}.$$

from Lemma 2.9. If $a \in \pi^n_{1+p}$, we have

$$1^T(v^{\pi^n} - v^{\pi^n_1}) = - \sum_{a \in \pi^n_{1+p}} x^{n+p}_{a} r^{\pi^n}_{a} \geq -x^{n+p}_{a_1} r^{\pi^n}_{a_1} \geq -r^{\pi^n}_{a_1},$$

where the first inequality is due to Lemma 2.10 and the second inequality is due to Lemma 2.9. Therefore, combining these two inequalities and the inequality (5) and noticing that $r^{\pi^n}_{a_1} \leq 0$, we get

$$-r^{\pi^n}_{a_1} \leq 1^T(v^{\pi^n} - v^{\pi^n_1}) < \frac{1 - \gamma}{l} 1^T(v^{\pi^n} - v^{\pi^n_1}) \leq - \frac{1}{1 - \gamma} \cdot \frac{l}{l} \cdot r^{\pi^n}_{a_1} = -r^{\pi^n}_{a_1}.$$

This leads to contradiction.

The previous derivation means that if $1^T(v^{\pi^n_{1+m+1}} - v^{\pi^n_{1+m}}) = 0$ does not hold for $n$, then an action of $\pi^n$ must disappear after $\pi^n_{1+m}$ forever. Hence, after every $M$ iterations, an action will disappear forever. This process cannot happen for more than $m - l$ times (since there are $m$ actions and every strategy has $n$ actions), which indicates that for some $n > M(m - l)$,

$$1^T(v^{\pi^n_{1+m+1}} - v^{\pi^n_{1+m}}) = 0.$$

It follows from the definition of geometrically converging algorithm that $\pi^{n+M}$ is the equilibrium strategy. This indicates that within $O(mM)$ number of iterations, we can find one of the equilibrium strategies.  

\[ \square \]
Our next theorem presents the complexity of the strategy iteration algorithm, the simplex strategy iteration algorithm and the modified simplex strategy iteration algorithm.

**Theorem 4.2:** The following algorithms have strongly polynomial convergence when the discount factor is fixed.

- The strategy iteration algorithm given in [7] can find the equilibrium strategy within \( O\left(\frac{m}{1 - \gamma}) \log\left(\frac{l}{1 - \gamma}) \right) \right) \) iterations.
- The simplex strategy iteration algorithm (Algorithm 1) can find the equilibrium strategy within \( O\left(\frac{ml}{1 - \gamma}) \log\left(\frac{l}{1 - \gamma}) \right) \right) \) iterations.
- The modified simplex strategy iteration algorithm (Algorithm 2) can find the equilibrium strategy within \( O\left(\frac{ml}{1 - \gamma}) \log\left(\frac{l}{1 - \gamma}) \right) \right) \) iterations.

**Proof:** The proof of this theorem directly follows from Theorem 4.1 and Proposition 3.3.

**Remark 4.1:** It is easy to note that the terminated condition of the simplex strategy iteration algorithm and the modified simplex strategy iteration algorithm is equivalent to the condition of meeting an equilibrium strategy. Hence, the above theorem also indicates that these two algorithms terminate within \( O\left(\frac{ml}{1 - \gamma}) \log\left(\frac{l}{1 - \gamma}) \right) \right) \) iterations.

**5. Conclusion**

In this paper, we propose a class of geometrically converging algorithms and develop a proof technique to prove the strongly polynomial complexity when the discount factor is fixed. The geometrically converging algorithms include the simplex strategy iteration algorithm and the modified simplex strategy iteration algorithm, and indicate the strongly polynomial property of these two algorithms. Specifically, our simplex strategy iteration algorithm is coincident with the simplex method in the MDP case. These analysis and properties shed some light on the open problem of solving the deterministic 2-TBSG in strongly polynomial time independent of the discount factor.

**Acknowledgments**

The authors are grateful to the associate editor and two anonymous referees for their valuable comments and suggestions.

**Disclosure statement**

No potential conflict of interest was reported by the authors.

**Funding**

The second author’s research was supported in part by the National Natural Science Foundation of China (NSFC) (grant number 11831002) and by the Beijing Academy of Artificial Intelligence.
Notes on contributors

Zeyu Jia, undergraduate student at Peking University. His research interests lie in reinforcement learning theory, machine learning theory and optimization. He is a member of the Elite Undergraduate Training Program of Applied Math in Peking University.

Zaiwen Wen, associate professor at Peking University. He holds a M.S. in Computational Mathematics from Academy of Mathematics and Systems Science (AMSS) (2004) and a Ph.D in Operations Research from Columbia University (2009). His research interests include large-scale computational optimization and their applications in sciences and engineering. He was awarded the National Natural Science Foundation of China for Excellent Young Scholars in 2013, the Young top-notch talent in 2015 and the Science and Technology Award for Chinese Youth in 2016. He is an associate editor of Journal of the Operations Research Society of China, Journal of Computational Mathematics and a technical editor of Mathematical Programming Computation.

Yinyu Ye, K.T. Li Chair Professor of Engineering at Department of Management Science and Engineering and Institute of Computational and Mathematical Engineering, Stanford University. He received a B.S. in System Engineering from the Huazhong University of Science and Technology, in China, and an MS and Ph.D in Engineering of Economic Systems and Operations Research from Stanford University. His current research interests include Continuous and Discrete Optimization, Data Science and Applications, Algorithm Design and Analysis, Computational Game/Market Equilibrium, Metric Distance Geometry, Dynamic Resource Allocation, and Stochastic and Robust Decision Making, among others. He has been a Fellow of INFORMS since 2012, and has received several academic awards, including the 2009 John von Neumann Theory Prize for fundamental sustained contributions to theory in Operations Research and the Management Sciences, the 2015 SPS Signal Processing Magazine Best Paper Award, the 2014 SIAG/Optimization Prize awarded every three years to the author(s) of the most outstanding paper, the inaugural 2012 ISMP Tseng Lectureship Prize for outstanding contributions to continuous optimization, and the inaugural 2006 Farkas Prize on Optimization. His doctoral students at Stanford have received the 2015 and 2013 Second Prize of the INFORMS Nicholson Student Paper Competition, the 2013 INFORMS Computing Society Prize, the 2008 First Prize of INFORMS Nicholson, and the 2006 and 2010 INFORMS Optimization Prizes for Young Researchers. He is the Chief Scientist of a major commercial optimization software company.

References

[1] R. Bellman, *Dynamic programming*, Science 153(3731) (1966), pp. 34–37.
[2] D.P. Bertsekas, D.P. Bertsekas, D.P. Bertsekas, and D.P. Bertsekas, *Dynamic Programming and Optimal Control*, Vol. 1, Athena Scientific, Belmont, MA, 1995.
[3] C. Derman, *Finite State Markov Decision Processes*, Academic Press, New York, 1970.
[4] J. Fearnley, *Exponential lower bounds for policy iteration*, in Proceedings of the Thirty-Seventh International Colloquium on Automata, Languages, and Programming, Springer, 2010, pp. 551–562.
[5] J. Filar and K. Vrieze, *Competitive Markov Decision Processes*, Springer Science & Business Media, New York, 2012.
[6] O. Friedmann, *An exponential lower bound for the parity game strategy improvement algorithm as we know it*, in Proceedings of the Twenty-Fourth Annual IEEE Symposium on Logic in Computer Science (LICS 2009), IEEE Computer Society Press, 2009, pp. 145–156.
[7] T.D. Hansen, P.B. Miltersen, and U. Zwick, *Strategy iteration is strongly polynomial for 2-player turn-based stochastic games with a constant discount factor*, J. ACM 60(1) (2013), pp. 1:1–1:16. ISSN 0004-5411.
[8] R.A Howard, *Dynamic Programming and Markov Processes*, MIT Press, Cambridge, 1960.
[9] M. Jurdziński and R. Savani, *A simple p-matrix linear complementarity problem for discounted games*, in *Logic and Theory of Algorithms*, A. Beckmann, C. Dimitracopoulos, and B. Löwe, eds., Springer, Berlin, 2008, pp. 283–293. ISBN 978-3-540-69407-6.
Appendix. Transform general 2-TBSGs into special 2-TBSGs

We prove in this section that every 2-TBSG can be transformed into a new 2-TBSG where each state has exactly two actions. A formal description is given in the next theorem.

**Theorem A.1:** Given a 2-TBSG with l states and m actions whose state set is \( S \), we can construct a new 2-TBSG with state set \( S' \) such that the following properties hold.

- The number of states in the constructed 2-TBSG is bounded by a polynomial of \( m \) and \( l \):
  \[ |S'| \leq m + l \log m = \tilde{O}(m + l). \]  
  (A1)

- \( S \subset S' \) and the value function \( V \) at the equilibrium of the constructed 2-TBSG satisfies
  \[ V(s) = c \cdot v(s) \quad \forall s \in S, \]  
  where \( v \) is the equilibrium value function of the original 2-TBSG, and
  \[ c = \gamma^\left(\lceil \log m \rceil - 1\right) / \lceil \log m \rceil. \]  
  (A2)

**Proof:** Our proof consists of two parts. In the first part, we construct a new 2-TBSG where each state has no more than two actions, and the value function at equilibrium of original 2-TBSG can be easily obtained given the equilibrium value of the constructed 2-TBSG (proportional to the value at some states in the constructed 2-TBSG). In the second part, we modify the constructed 2-TBSG so that each state has exactly two actions, while keeping the equilibrium value unchanged by constructing an obvious undesirable action for those states with only one action.

We first construct a binary tree rooted at \( s \) with exactly \(|A_s|\) leaves, and the depth of the tree is exactly \( p = \lceil \log m \rceil \). This tree is called the depth-\( p \) binary tree of state \( s \):

- In the first \( p - \log |A_s| \) layers, each node has only one child.
- In the last \( \log |A_s| \) layers, it is a binary tree with exactly \(|A_s|\) leaves.
• Every leaves has depth $p$.

Each node except the root $s$ and all leaves in the depth-$p$ binary tree of $s$ are assigned with a new state whose owner is same as state $s$ (player 1 or player 2). We use $S_1$ to denote the set of states in the first $p-2$ layers and $S_2$ to denote the set of states in the $(p-1)$-th layer. The parameters (transition probabilities, rewards, discount factor) are given as follows:

• For each state in $S_1$, one or two actions are assigned to it depending on how many children states (its children in the binary tree) it has, with probability 1 leading to a child state and reward 0.

• For set $S_2$, each of their children nodes is assigned with an action of $s$ in the original 2-TBSG. This can be done since the total number of children nodes of $S_2$ is exactly $|A_s|$. For each state in $S_2$, its actions are given by its children nodes. The transition probability and reward of taking that action is assigned to be the same as in the original 2-TBSG.

• The discount factor in the constructed 2-TBSG is given by $\delta = \gamma^{1/p}$.

A special case can be viewed in Figure A1 when $p = 4$, $|A_s| = 7$. It is easy to obtain that the number of states in the constructed 2-TBSG is no more than $m + l \log m$.

We next present a definition of final actions and the executing path of a state.

**Definition A.2.** For a given strategy $\pi'$ in the constructed 2-TBSG cases and $s \in S$, we continue the following process:

• $s_0 \leftarrow s$, $i \leftarrow 0$.

• If $\pi'(s_n)$ is a constructed action (not an action in the original 2-TBSG), then we let $s_{n+1}$ to be the state obtained by executing action $\pi'(s_n)$. Since all constructed actions are deterministic, there is only one choice of $s_{n+1}$. Then let $n \leftarrow n + 1$.

• If $\pi'(s_n)$ is an action in the original 2-TBSG, then we stop this process, and call $\pi'(s_n) \in A$ to be the final action of $s$, and path $s_0 \rightarrow s_1 \rightarrow \cdots \rightarrow s_n \rightarrow \pi'(s_n)$ to be the executed path from $s$ to action $\pi'(s_n)$.

Notice that the previous described process must be ended in $p-1$ steps, and all states in the executed path of $s$ must lie in the depth-$p$ binary tree of $s$.

For any state $s \in S$ and $a \in A_s$, there exists a unique executed path from $s$ to $a$. In Figure A1, we present an example of final actions and executed

```
\[ s \quad s_1 \quad s_2 \quad s_3 \]
\[ s_4 \quad s_5 \quad s_6 \quad s_7 \]
\[ a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6 \quad a_7 \]
```

**Figure A1.** Example when $t = 4$. 
paths. When the strategy \( \pi' \) follows bold arrows, the final action of \( s \) will be \( a_3 \), and the executed path from \( s \) to \( a_3 \) is \( s \rightarrow s_1 \rightarrow s_2 \rightarrow s_5 \rightarrow a_3 \).

Based on the final actions, we define the corresponding strategy \( \pi \) with respect to \( \pi' \) in the original 2-TBSG: for each state \( s \in S \), \( \pi(s) \) is defined to be the final action of \( s \) in \( \pi' \). Next, we prove that for any state \( s \in S \), the value of \( s \) in strategy \( \pi' \) agrees with \( \delta^{p-1} \) times the value of \( s \) in strategy \( \pi \). Actually, along the trajectory of \( \pi' \), we meet a final action every \( p \) steps, and only final actions have nonzero rewards. Hence, values of \( s \) in \( \pi \) and \( \pi' \) satisfy

\[
V^{\pi'}(s) = \mathbb{E}_{\pi'} \sum_{i=1}^{\infty} r_a \cdot \delta^{p-1-i} = \delta^{p-1} \mathbb{E}_{\pi} \sum_{i=1}^{\infty} r_a \cdot \gamma^{i-1} = \delta^{p-1} V^{\pi}(s),
\]

where \( a' \) denotes actions along strategy \( \pi' \), and \( a \) denotes actions along strategy \( \pi \).

What is left in the proof is to show that if \( \pi' = (\pi'_1, \pi'_2) \) is an equilibrium strategy in the constructed 2-TBSG, then \( \pi = (\pi_1, \pi_2) \) is an equilibrium strategy in the original 2-TBSG. For any player 1’s state \( s \) and action \( a \in A_s \), we use \( \eta = (\eta_1, \pi_2) \) to denote the strategy in the original 2-TBSG:

\[
\eta_1(s_1) = \begin{cases} a & \text{if } s_1 = s, \\ \pi_1(s_1) & \text{otherwise}. \end{cases}
\]

In the constructed 2-TBSG, there exists a unique executed path \( P \) from \( s \) to action \( a \), and for any state \( s_1 \) on this path \( P \), there is only one action \( \tau(s_1) \) in \( A'_1 \), such that the next state when using \( \tau(s_1) \) also lies on \( P \). We define player 1’s strategy \( \eta' = (\eta'_1, \pi_2) \) as follows:

\[
\eta'_1(s_1) = \begin{cases} \tau(s_1) & \text{if } s_1 \in P, \\ \lambda(s_1) & \text{if } s_1 \text{ is in the depth-} \cdot \text{binary tree of } s \text{ and } s_1 \notin P, \\ \pi'_1(s_1) & \text{if } s_1 \text{ is not in the depth-} \cdot \text{binary tree of } s, \end{cases}
\]

where \( \lambda(s_1) \) can be chosen \( A'_1 \) arbitrarily. Then, it is easy to examine that \( \eta \) is the corresponding strategy of \( \eta' \). Since \( \eta \) is an equilibrium strategy of the constructed 2-TBSG, we have

\[
V^\eta = \delta^{-p-1} V^{\eta'} \leq \delta^{-p-1} V^{\pi'} = V^\pi,
\]

where the inequality is due to the property of equilibrium strategy. Furthermore, according to Lemmas 2.9 and 2.10, we have \( x^\eta_1 r_1^\pi = \mathbb{I}^T (V^\eta - V^\pi) \leq 0 \), where \( x^\eta_1 \geq 1 \). This indicates that \( r_1^\pi \leq 0 \).

If we construct actions in such ways, it is obvious that action \( a_2 \) is inferior to \( a_1 \) according to its owner (player 1 or player 2). Hence, any strategy which possesses \( a_2 \) is not an equilibrium strategy, since switching action \( a_2 \) into \( a_1 \) leads to a better strategy for its owner. Combining these two parts together proves Theorem A.1.

**Remark A.1:** Since it is easy to obtain the equilibrium strategy from the equilibrium value and vice versa, we can solve the original 2-TBSG by solving the constructed 2-TBSG.