Finding Nash Equilibria for Polymatrix Games

Lunshan (Shaun) GAO

Engineering Department, Raytheon Canada Limited, 400 Phillip Street, Waterloo, ON. N2L6R7, Canada

lunshan.gao@gmail.com

***Abstract*-This paper describes the derivation of the expected payoff function of polymatrix games according to the induction method. It also presents a new algorithm for calculating mixed Nash equilibrium (NE) in polymatrix games. Results indicate that the new algorithm can compute mixed NEs for polymatrix games within polynomial time. This paper is a continuation result of previous research which describes that the expected payoff function of 2-player games in normal form is identical to the mathematical representation of the fuzzy average of two linguistic values of a linguistic variable; this paper extends the identification of 2-player games to polymatrix games.**

Keywords- N-player Non-cooperative Game in Normal Form; Polymatrix Games; Nash Equilibrium; Expected Payoff Function; Fuzzy Average; Linguistic Variables; Triangular Fuzzy Number

I. INTRODUCTION

A game in normal form, also known as strategic game, is a static model which describes interactive situations among several players. According to this model, for non-cooperative games, all players make their decisions simultaneously and independently. A polymatrix game is a particular class of n-player game in normal form, in which the payoff for each player is obtained as a sum of individual payoffs gained against each other player.

Finding NEs in game theory is a fundamental problem. For two-player games, the Lemke-Howson algorithm [1] is still state-of-the-art, despite its 50 year age. Howson [2] extended the Lemke-Howson algorithm to polymatrix games, positing that polymatrix games are linear combinations of two-player games, which have linear structures. However, the Lemke-Howson algorithm cannot be directly applied to other games due to the fact that NE is no longer a linear complementarity problem when n is greater than two. Rosenmuller [3] and Wilson [4] independently extended the Lemke-Howson algorithm to determine NEs for n-player games.

A different method to compute NEs was introduced by Scarf [5], called the simplicial subdivision algorithm. This algorithm is widely used to determine NEs for n-player games. Dickhaut and Kaplan [6] proposed a search algorithm which enumerated all support profiles in order to determine all NEs. Govindan and Wilson [7] proposed another method based on the global Newton method combined with homotopy to compute NEs for n-player games. Govindan and Wilson [8] published another new method which merges the Lemke-Howson algorithm with the global Newton method. Blum, et al., [9] improved the method which was proposed by Govindan and Wilson [8] and applied it to both graphical games and multi-agent influence diagrams. Porter, et al., [10] presented two algorithms for finding NEs: one for two-player games and one for n-player games. Cai, et al., [11] demonstrated that calculating NEs for zero-sum polymatrix games is equivalent to solving linear programming, and showed that von Neumann’s minimax theorem for two-player zero-sum games can be generalized to polymatrix games. Belhaiza [12] demonstrated that every perfect equilibrium of a polymatrix game is un-dominated, and that every un-dominated equilibrium of a polymatrix game is perfect.

A recent sequence of papers indicates that computing one (any) NE is PPAD (Polynomial Parity Arguments on Directed graphs)-complete for two-, three-, or four-player games in strategic form [13-16]. Chen and Deng [14] demonstrated that computing NEs for two-player games is PPAD-complete. Daskalakis, et al., [15] showed that determining NEs for four-player games is PPAD-complete. Chen and Deng [13] and Daskalakis and Papadimitriou [16] independently demonstrated that calculating NEs for three-player games is PPAD-complete. All known algorithms require exponential time in the worst case. Chen, et al., [17] reported that the problem of computing a (1/n)-well-supported NE in a polymatrix game is PPAD-complete.

Fuzzy theory has been applied to game theory [18-20]. Chakeri and Sheikholeslam [18] proposed a method of determining fuzzy NEs in crisp and fuzzy games. Garagic and Cruz [19] extended the concept of non-cooperative game theory to fuzzy non-cooperative games under uncertainty phenomena. Wu and Soo [20] applied fuzzy game theory to multi-agent coordination. In this paper, fuzzy theory is used as a tool to reduce the complexity of calculating NEs in polymatrix games.

Results:

1. This article describes the derivation of expected payoff function in polymatrix games, and presents the reason why the expected payoff of each player in non-cooperative games is obtained as a sum of individual payoffs gained against each other player.
2. Associated with the derivation, a new algorithm for computing NEs for polymatrix games is presented. The algorithm’s ability to calculate NEs in polymatrix games within polynomial time is also demonstrated.

DOI: 10.5963/IJCSAI0501001

This article is organized as follows. Section 2 presents preliminaries which include the definitions of n-player games, fuzzy numbers, and the fuzzy average. Section 3 describes the formula of the expected payoff function for n-player polymatrix games, and a theorem for computing mixed NEs. Section 4 describes the proposed algorithm in detail. Section 5 presents three examples, and section 6 presents conclusions and ideas for future work.

II. PRELIMINARIES

Below are described some basic concepts in game theory and fuzzy set theory.

A. The Strategic Form of an n-player Game

An n-player game in strategic form: an n-player game in strategic form is described by a tuple $G = (N,\ \{S_i\}_{i \in N},\ \{u_i\}_{i \in N})$ with the following:

(1) A finite set of players $N = \{1, 2, \ldots, n\}$.

(2) A set $S_i = (s_{i1}, \ldots, s_{ik})(i = 1,2,..n)$ of strategies for each player $i \in N$. Without losing generality, suppose that all players have the same number of strategies in this paper.

(3) A utility function $u_i : S \to R$ for each player, where $S = \prod_{i \in N} S_i$ is the space of pure strategies.

The utility function u_i is a real value function which maps the space of all players' strategies into a real value.

Best Response: Let $S_{-i} := (S_1, \ldots, S_{i-1}, S_{i+1}, ..S_n)$. Player *i*'s best response is a mixed strategy $S_i^* \in S$ such that $u_i(S_i^*, S_{-i}) \geq u_i(S_i, S_{-i})$ for all strategies $S_i \in S$.

Nash Equilibrium: A strategy profile $S = (S_1, S_2, \ldots, S_n)$ is an NE if for all players *i*, S_i represents the best response to S_{-i}.

B. Fuzzy Numbers

A fuzzy number is a fuzzy set which is defined in *R*. There some various types of fuzzy numbers [21], including triangular fuzzy numbers (TFNs), trapezoidal fuzzy numbers, etc. TFNs only are reviewed here.

A TFN is denoted as *(a, b, c)*, where $a \in R; b \in R; c \in R$ $(a \leq b \leq c)$, and the membership function of TFN *(a, b, c)* is described as follows.

$$\mu(x) = \begin{cases} \dfrac{x-a}{b-a} & if \quad x \in [a,b] \\ \dfrac{c-x}{c-b} & if \quad x \in (b,c] \\ 0 & otherwise \end{cases}$$

There are two special TFNs. One is *(a, a, c)*, such that *a* = *b*; the other is *(a, c, c)*, such that *b* = *c*. The membership functions of TFN *(a, a, c)* and TFN *(a, c, c)* are described as follows.

$$\mu(x) = -\frac{1}{c-a}x + \frac{c}{c-a}\ ; (a < c),\ \ \nu(x) = \frac{1}{c-a}x - \frac{a}{c-a}\ ; (a < c).$$

C. The Fuzzy Average

The fuzzy average [22] is defined as the average of linguistic values of a linguistic variable *(x, T(x), U, G, M)* [23, 24], where *x* is the name of the variable; *T(x)* is the term set of *x*; *U* is the universe of discourse, which is usually defined as the interval [0, 1]; *G* is the syntactic rule which generates the terms in *T(x)*; *M* is a semantic rule, which is typically a mapping from the set *T(x)* to a set of fuzzy numbers defined in *U*. The fuzzy average of two values of a linguistic variable is described as follows.

Suppose that the two values of a linguistic variable are $(x, T_1(x), U_1, G_1, M_1)$ and $(y, T_2(y), U_2, G_2, M_2)$, where $T_1(x) = \{x_1, x_2, \ldots, x_n\}$, $T_2(y) = \{y_1, y_2, \ldots, y_m\}$; $U_1 = [0,\ \ 1]$; $U_2 = [0,\ \ 1]$.

DOI: 10.5963/IJCSAI0501001

$M_1 : T_1(x) \to \{A_1, A_2, \ldots, A_n\}$, $M_2 : T_1(y) \to \{B_1, B_2, \ldots, B_m\}$, where A_i (i=1,2,…,n) and B_j (j=1, 2,…,m) are triangular fuzzy numbers (TFNs) which are defined on $U_1 = [0,\ 1]$; $U_2 = [0,\ 1]$, $\mu_{A_i}(x)$ and $\mu_{B_j}(y)$ are the membership functions of TFNs A_i and B_j, respectively. The fuzzy average is defined as:

$$u(x, y) = \sum_{i=1}^{n}\sum_{j=1}^{m} \mu_{A_i}(x) \times \mu_{B_j} \times r_{ij} = \mu_A(x) \times R \times \mu_B(y)$$

where

$\mu_A(x) = \{\mu_{A_1}(x_1), \mu_{A_2}(x_1), ..\mu_{A_n}(x_1)\}$, $\mu_A(x) \in \Delta_n(\mu_A(x))$;

$\mu_B(y) = \{\mu_{B_1}(y_1), \mu_{B_2}(y_1), ..\mu_{B_m}(y_1)\}$, $\mu_B(y) \in \Delta_m(\mu_B(y))$;

$\Delta_n(z) := \{z_i \mid \sum_{i=1}^{n} z_i = 1; z_i \geq 0 (i = 1,2..n)\}$, $x \in U_1$ and $y \in U_2$;

n is the number of entries in $T_1(x)$; m is the number of entries in $T_2(y)$; and $R = (r_{ij})$ is the consequence matrix [22]. It has been proven that the fuzzy average converges to the arithmetic mean [22]. $\mu_{A_i}(x)$ is interpreted as the weight of element $x_i \in T_1(x)$. For a given $x \in U_1$ and $y \in U_2$, the vector $\mu_A(x)$, $\mu_B(y)$ is interpreted as the probability distribution over $T_1(x)$, $T_2(y)$, respectively.

In game theory, the set of strategies $S_i = (s_{i1}, \ldots, s_{ik})(i = 1,2,..n)$ can be interpreted as the term set $T(S_i)$ in the concept of linguistic variables. For example, for a rock-paper-scissors game, a player's strategy set $S = (r, p, s)$ (where r, p, and s stand for rock, paper and scissors, respectively) can be considered as a term set of $\{r,\ p,\ s\}$ in linguistic variables, such that $T_1(S) = \{r, p, s\}$ and $T_2(S) = \{r, p, s\}$. The probability distribution over the set of strategies for each player can be represented by $\mu_A(x)$ and $\mu_B(y)$.

For two-player games in normal form, when a player's strategy set is represented by the term set and the payoff matrix is the same as the consequence matrix, it was proved that the expected payoff function is identical to the fuzzy average [25, 26].

III. EXPECTED PAYOFF FUNCTION OF POLYMATRIX GAMES

The derivation of the expected payoff function of polymatrix games is described in this section.

For an n-player non-cooperative game $G = (N,\ \{S_i\}_{i \in N},\ \{u_i\}_{i \in N})$, player i's strategy set $S_i = (s_{i1}, ..., s_{ik})$ corresponds to a probability distribution P_i over S_i, where $P_i = (p_{i1}, p_{i2}, \ldots p_{ik})$, $P_i \in \Delta_k(P_i)$; $(i = 1,2..n)$. For example, for two players in a rock-paper-scissors game, strategy set $S_i = (r_i, p_i, s_i)(i = 1,2)$ for each player. A mixed strategy is described by probability distributions; for instance $P_1 = (1/2, 1/4, 1/4)$ represents a strategy over the strategy set S_1. The probability that player 1 chooses rock, paper, scissors is 0.5, 0.25 and 0.25, respectively. When P_1 is a unit vector, then the mixed strategy becomes pure strategy. To summarize, the strategy set S_i always corresponds to a probability distribution P_i over the strategy.

In this paper, the expected payoff function is described by probability distributions rather than using strategy sets. That is, the expected payoff function is defined as a map from a set of probability distribution functions (PDFs) over S to a real value. For example, a two-player game in normal form given by $G = (2,\ \{S_i\}_{i \in 2},\ \{u_i\}_{i \in 2})$, where $S_i = (s_{i1}, \ldots, s_{ik})(i = 1,2)$ is a set of strategies for player i; $u_i(i = 1,2)$ is the expected payoff function. The function is described as follows.

$$u_1(P_1, P_2) = P_1 \times A_{12} \times P_2^T$$

$$u_2(P_2, P_1) = P_2 \times A_{21} \times P_1^T$$

where $P_i = \{p_{i1}, \ldots, p_{ik}\} \in \Delta_k(P_i); (i = 1,2)$ represents the probability distribution over S_i; A_{12}, A_{21} is the $k \times k$ payoff matrix for player 1 and player 2, respectively.

A_{12} and A_{21} are also known as a bi-matrix. The first number in the index of a payoff matrix indicates to which player the payoff matrix belongs. The order of the indices of A_{12} and A_{21} indicates which player competes with which player. For example, A_{12} is player 1's payoff matrix by competing with player 2; A_{21} is player 2's payoff matrix by competing with player 1.

DOI: 10.5963/IJCSAI0501001

Suppose a new player, denoted as player 3 with strategy set S_3, is added into the game. Associated with S_3, a probability distribution over S_3 is denoted as P_3.

Since the game is non-cooperative, each player is an independent individual. Player 3 must compete with each existing player and vice versa. As a result, the following four combinations are formed.

$$Player\ \ 1 \rightarrow Player\ \ 3,\ \ Player\ \ 2 \rightarrow Player\ \ 3,\ \ Player\ \ 3 \rightarrow Player\ \ 1,\ \ Player\ \ 3 \rightarrow Player\ \ 2.$$

Assume that A_{13}, A_{23}, A_{31} and A_{32} are the payoff matrices representing the above four competitions. Then one can obtain the following payoff sub-functions.

$$u_{13}(P_1, P_3) = P_1 \times A_{13} \times P_3^{\ T},\ \ u_{23}(P_2, P_3) = P_2 \times A_{23} \times P_3^{\ T},$$

$$u_{31}(P_3, P_1) = P_3 \times A_{31} \times P_1^{\ T},\ \ u_{32}(P_3, P_2) = P_3 \times A_{32} \times P_2^{\ T}.$$

In order to obtain the expected payoff function for the three-player game, rewrite the expected payoff functions $u_1(P_1, P_2)$ and $u_2(P_2, P_1)$ from the original two-player game as follows.

$$u_{12}(P_1, P_2) = P_1 \times A_{12} \times P_2^{\ T},$$

$$u_{21}(P_2, P_1) = P_2 \times A_{21} \times P_1^{\ T}.$$

Since the game is non-cooperative, each player makes decision independently. The expected payoff function for each player becomes as follows.

$$u_1(P_1, P_2, P_3) := u_1(P_1, P_{-1}) = u_{12}(P_1, P_2) + u_{13}(P_1, P_3) = P_1 \times A_{12} \times P_2^{\ T} + P_1 \times A_{13} \times P_3^{\ T}$$

$$u_2(P_2, P_1, P_3) := u_2(P_2, P_{-2}) = u_{21}(P_2, P_1) + u_{23}(P_2, P_3) = P_2 \times A_{21} \times P_1^{\ T} + P_2 \times A_{23} \times P_3^{\ T}$$

$$u_3(P_3, P_1, P_2) := u_3(P_3, P_{-3}) = u_{31}(P_3, P_1) + u_{32}(P_3, P_2) = P_3 \times A_{31} \times P_1^{\ T} + P_3 \times A_{32} \times P_2^{\ T}$$

If the above procedure is repeated, one can obtain the following theorem.

Theorem 3.1

Given an n-player non-cooperative game $G = \{N, \{S_i\}_{i \in N}, \{u_i\}_{i \in N}\}$, where $S = \prod_{i \in N} S_i$, $S_i = (s_{i1}, \ldots, s_{ik})(i = 1,2,..n)$, $u = \{u_1, u_2, \ldots, u_N)$ is a vector of expected payoff functions of the n-player games. If $P_i(i = 1,2,..n)$ is player *i*'s probability distribution over S_i, where $P_i = \{p_{i1}, p_{i2}, \ldots, p_{ik}\}$ $P_i \in \Delta_k(P_i), (i = 1,2,..n)$, then the expected payoff function for player *i* takes the following form.

$$u_i(P_i, P_{-i}) = \sum_{j=1}^{i-1} P_i \times A_{ij} \times P_j^{\ T} + \sum_{j=i+1}^{N} P_i \times A_{ij} \times P_j^{\ T}, (i = 1,2,\ldots N) \tag{1}$$

where $A_{ij}(i, j = 1,2,\ldots,N;\ j \neq i)$ is a $k \times k$ matrix (*k* is the number of strategies), which represents player *i*'s payoff matrix.

The induction method is utilized to prove theorem 3.1.

Proof:

When *n* = 2,

$u_1(P_1, P_2) = P_1 \times A_{11} \times P_2^{\ T}$

$u_2(P_2, P_1) = P_2 \times A_{21} \times P_1^{\ T}$

These are exactly the expected payoff functions of two-player games in normal form.

Suppose that the Formula (1) holds when n=L, such that:

$$u_i(P_i, P_{-i}) = \sum_{j=1}^{i-1} P_i \times A_{ij} \times P_j^{\ T} + \sum_{j=i+1}^{L} P_i \times A_{ij} \times P_j^{\ T}(i = 1,2,\ldots L) \tag{2}$$

DOI: 10.5963/IJCSAI0501001

One player is added with strategy set S_{L+1}, and its probability distribution P_{L+1} over S_{L+1}. Then, the game becomes an L+1 player game. Because the game is non-cooperative game, there are 2L combinations between the new player and the existing L players, such that:

$player \ \ i \rightarrow player \ \ L+1 \ \ (i=1,2,..L) \Rightarrow$ payoff matrix $A_{iL+1}(i=1,2,..L)$ and $player \ \ L+1 \rightarrow player \ \ i \ \ (i=1,2,..L) \Rightarrow$ payoff matrix $A_{L+1i}(i=1,2,..L)$.

As a result, a total of 2L payoff sub-functions are generated.

$$u_{iL+1}(P_i, P_{L+1}) = P_i \times A_{iL+1} \times P_{L+1}{}^T, \qquad (i=1,2,..L) \tag{3}$$

$$\begin{aligned} u_{L+1}(P_{L+1}, P_i) &= u_{L+11}(P_{L+1}, P_1) + u_{L+12}(P_{L+1}, P_2) + \ldots + u_{L+1L}(P_{L+1}, P_L) \\ &= \sum_{j=1}^{L} u_{L+1j}(P_{L+1}, P_j) = \sum_{j=1}^{L} P_{L+1} \times A_{L+1j} \times P_j{}^T \end{aligned} \tag{4}$$

For each existing player, the expected payoff function becomes (2) + (3), such that:

$$u_i(P_i, P_{-i}) = \sum_{j=1}^{i-1} P_i \times A_{ij} \times P_j{}^T + \sum_{j=i+1}^{L} P_i \times A_{ij} \times P_j{}^T + P_i \times A_{iL+1} \times P_{L+1}{}^T (i=1,2,\ldots L) \tag{5}$$

Therefore, from (4) and (5), the following is obtained.

$$u_i(P_i, P_{-i}) = \sum_{j=1}^{i-1} P_i \times A_{ij} \times P_j{}^T + \sum_{j=i+1}^{L+1} P_i \times A_{ij} \times P_j{}^T, \qquad (i=1,2,\ldots L+1) \tag{6}$$

Formula (6) indicates that when $N = L+1$, Formula (1) holds.

As a result, Formula (1) represents player i's expected payoff function for an n-player polymatrix game.

Theorem 3.1 provides a general formula of the expected payoff function for every player in an n-player polymatrix game in normal form. According to Formula (1), the following theorem is obtained.

Theorem 3.2

For an n-player polymatrix game $G = (N, \ \{S_i\}_{i \in N}, \ \{u_i\}_{i \in N})$, if:

(1) Formula (1) is piecewise continuously differential regarding P_i, then

(2) Equation (7) has finite solutions.

$$\frac{\partial u_i(P_i, P_{-i})}{\partial P_i} = 0, (i=1,2,\ldots N) \tag{7}$$

(3) The solution $P_i = \{p_{i1}, p_{i2}, \ldots, p_{ik}\}$ satisfies $P_i \in \Delta_k(P_i), (i=1,2,..N)$.

Therefore, the n-player polymatrix game has mixed NEs, and the solutions of (7) represent the mixed NEs of the n-player polymatrix game.

Two steps are required to prove this theorem. First, Equation (7) has become a system of linear equations when PDF P_i is replaced by certain semantic rules. Second, the solutions of (7) are proved to be mixed NEs.

Proof:

Semantic rules are defined to represent the probability distributions P_i and P_{-i} in (1), rather than using exponential PDFs to describe them.

$$P_i = \mu_{P_i}(x) = \{\mu_{P_{i1}}(x), \mu_{P_{i2}}(x), \ldots \mu_{P_{ik}}(x)\} \tag{8}$$

$$P_{-i} = \mu_{P_{-i}}(y) = \{\mu_{P_{-i1}}(y), \mu_{P_{-i2}}(y), \ldots \mu_{P_{-ik}}(y)\} \tag{9}$$

where $P_i \in \Delta_k(P_i) \ (i,=1,2,..n)$; $\mu_{P_i}(x), \mu_{P_{-i}}(y)$ is the vector of the membership functions of TFNs, which represents the probability distributions over S_i and S_{-i} respectively.

Since the payoff function is a real value, one obtains the following,

DOI: 10.5963/IJCSAI0501001

$$\frac{\partial u_i(P_i, P_{-i})}{\partial P_i} = \sum_{j=1}^{i-1} \frac{d\mu_{P_i}(x)}{dx} \times A_{ij} \times \mu_{P_{-i}}(y) + \sum_{j=i+1}^{N} \frac{d\mu_{P_i}(x)}{dx} \times A_{ij} \times \mu_{P_{-i}}(y) = 0 \quad (10)$$

Based on the properties of TFNs, $\frac{d\mu_{P_i}(x)}{dx}$ is a constant vector, and $\mu_{P_{-i}}(y)$ is a vector of the piecewise linear functions of y. Therefore, Equation (10) becomes a system of linear equations when the elements of payoff matrix A_{ij} are constant.

Based on condition (2), the linear system has finite solutions. If (1) is proven to be a concave function regarding P_i, then the solution of Equation (7) represents the maximum value of (1). As a result, the solution of Equation (7) is an NE.

In order to prove that (1) is a concave function, the following inequality must satisfy for $t \in [0, \ 1]$.

$$u_i((1-t)P_i + tP_i, P_{-i}) \geq (1-t)u_i(P_i, P_{-i}) + tu_i(P_i, P_{-i})$$

In fact, one can obtain the following from (1):

$$u_i((1-t)P_i + tP_i, P_{-i}) = (1-t)u_i(P_i, P_{-i}) + tu_i(P_i, P_{-i}) \quad (11)$$

This indicates that the function $u_i(P_i, P_{-i})$ is a concave function regarding P_i. Therefore, the solution is a mixed NE of the polymatrix game.

The proof of theorem 3.2 provides the following proposition.

Proposition:

Given a polymatrix game $G = (N, \ \{S_i\}_{i\in N}, \ \{u_i\}_{i\in N})$, the expected payoff function is described by Formula (1), and the probability distributions are defined as (8) and (9); then, computing mixed NEs in the polymatrix games can be achieved in polynomial time by solving (7).

Proof:

As described in the proof of theorem 3.2, when PDFs P_i and P_{-i} are replaced with (8) and (9), Equation (10) becomes a system of linear equations. Because solving a system of linear equations can be completed within polynomial time [27], therefore, solving equation (10) can be completed within polynomial time. Thus, computing mixed NEs in polymatrix games can be completed in polynomial time by using the proposed algorithm.

IV. THE ALGORITHM OF COMPUTING MIXED NES IN POLYMATRIX GAMES

The new algorithm is an extension of the algorithm for two-player games [25, 26]. The basic concept is the relationship between the expected payoff function of two-player games in normal form and the concept of the fuzzy average. It was proved that the expected payoff function of two-player games in normal form is identical to the fuzzy average of two linguistic values when the strategy sets in two-player games are represented with the term sets in linguistic variables; the payoff matrix is replaced with the consequence matrix, and the probability distribution over the strategy set for each player is represented with the semantic rule *M* in linguistic variables [25]. The algorithm to calculate mixed NEs in polymatrix games is described as follows.

1. Given an n-player polymatrix game $G = (N, \ \{S_i\}_{i\in N}, \ \{u_i\}_{i\in N})$ in normal form, the following steps build n linguistic values $(S_i, T(S_i), U_i, G_i, M_i)(i \in N)$.

   i) Define the term sets by using the strategy sets, such as $T(S_i) = \{S_i\}, (i \in N)$.

   ii) Define $U_i = [0, \ 1]$.

   iii) Divide domain U_i into k - m $(m < k)$ partitions, where k is the number of strategies, such as $U_i = \{U_{i1}, U_{i2}, \ldots, U_{ik-m}\}$.

   iv) Define proper semantic rules $M_i : T(S_i) \to P_i (i \in N)$, where P_i is defined as (8).

2. Construct the expected payoff function (1) using the given payoff matrices, and the probability distributions (8) and (9).
3. Make $(k-m)^N$ combinations of domains by combining each sub-domain in U_i for all players, such as

$D_1 = U_{11} \times U_{21} \times \ldots \times U_{n1}$, $D_2 = U_{11} \times U_{21} \times \ldots \times U_{n2}$, ……, $D_{k-m} = U_{11} \times U_{21} \times \ldots \times U_{nk-m}$,

$D_{k-m+1} = U_{11} \times U_{21} \times \ldots \times U_{n-12} \times U_{n1}$, $D_{k-m+2} = U_{11} \times U_{21} \times \ldots \times U_{n-12} \times U_{n2}$, ……,

………

DOI: 10.5963/IJCSAI0501001

$D_{(k-m)^N} = U_{1k-m} \times U_{2k-m} \times \ldots \times U_{n-1k-m} \times U_{nk-m}$.

4. Solve (7) in each domain $D_j\,(j = 1,2,\ldots,(k-m)^N)$.
5. Verify that the solution of (7) belongs to its domain. If the solution is verified, then accept the solution; otherwise, ignore it.

V. EXAMPLES

This section provides three examples to verify the proposed algorithm.

Example 1. Find mixed NEs in the three-player matching of pennies.

	Player 3 takes head			Player 3 takes tail	
	Player 2 takes head	Player 2 takes tail		Player 2 takes head	Player 2 takes tail
Player 1 takes head	0, 0, 0	1, -2, 1	Player 1 takes head	1, 1, -2	-2, 1, 1
Player 1 takes tail	-2, 1, 1	1, 1, -2	Player 1 takes tail	1, -2, 1	0, 0, 0

Player 1's payoff matrices are described by $A_{12} = \begin{pmatrix} 0, & 1 \\ -2, & 1 \end{pmatrix}$, $A_{13} = \begin{pmatrix} 1, & -2 \\ 1, & 0 \end{pmatrix}$

Player 2's payoff matrices are described by $A_{21} = \begin{pmatrix} 0, & -2 \\ 1, & 1 \end{pmatrix}^T = A_{12}$, $A_{23} = \begin{pmatrix} 1, & 1 \\ -2, & 0 \end{pmatrix}^T = A_{13}$

Player 3's payoff matrices are described by $A_{31} = \begin{pmatrix} 0, & 1 \\ 1, & -2 \end{pmatrix}$, $A_{32} = \begin{pmatrix} -2, & 1 \\ 1, & 0 \end{pmatrix}$

The expected payoff function of each player is as follows.

$u_1(P_1, P_{-1}) = P_1 \times A_{12} \times P_2^{\ T} + P_1 \times A_{13} \times P_3^{\ T}$

$u_2(P_2, P_{-2}) = P_2 \times A_{21} \times P_1^{\ T} + P_2 \times A_{23} \times P_3^{\ T}$

$u_3(P_3, P_{-3}) = P_3 \times A_{31} \times P_1^{\ T} + P_3 \times A_{32} \times P_2^{\ T}$

The semantic rule M is defined as follows.

The probability distributions P_1, P_2 and P_3 over $S_1 = S_2 = S_3$ = {head, tail} are defined as follows.

$$P_1 = (B_{11}, B_{21}),\ P_2 = (B_{12}, B_{22}),\ P_3 = (B_{13}, B_{23})$$

where $B_{11} = B_{12} = B_{13}$ = (0, 1, 1); $B_{21} = B_{22} = B_{23}$ = (0, 0, 1) are TFNs defined in domain $[0,1]\times[0,1]\times[0,1]$. According to theorem 3.2, one can obtain the following.

$$\frac{\partial u_1(P_1,P_{-1})}{\partial P_1} = (\frac{dB_{11}}{dx}, \frac{dB_{21}}{dx}) \times \left[\begin{pmatrix} 0 & 1 \\ -2 & 1 \end{pmatrix} \times \begin{pmatrix} y \\ 1-y \end{pmatrix} + \begin{pmatrix} 1 & -2 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} z \\ 1-z \end{pmatrix} \right] = 0 \tag{12}$$

$$\frac{\partial u_2(P_2,P_{-2})}{\partial P_2} = (\frac{dB_{12}}{dy}, \frac{dB_{22}}{dy}) \times \left[\begin{pmatrix} 0 & 1 \\ -2 & 1 \end{pmatrix} \times \begin{pmatrix} x \\ 1-x \end{pmatrix} + \begin{pmatrix} 1 & -2 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} z \\ 1-z \end{pmatrix} \right] = 0 \tag{13}$$

$$\frac{\partial u_3(P_3,P_{-3})}{\partial P_3} = (\frac{dB_{13}}{dz}, \frac{dB_{23}}{dz}) \times \left[\begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix} \times \begin{pmatrix} x \\ 1-x \end{pmatrix} + \begin{pmatrix} -2 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} y \\ 1-y \end{pmatrix} \right] = 0 \tag{14}$$

Equations (12), (13) and (14) form a linear system with variables x, y and z as follows.

$$\begin{cases} \quad\ \ y + z = 1 \\ x + \quad\ \ z = 1 \\ x + y \quad\ \ = 1 \end{cases}$$

It has unique solution $(x, y, z) = (1/2, 1/2, 1/2)$.

DOI: 10.5963/IJCSAI0501001

Therefore, this three-player matching of pennies has a mixed NE (s_1^*, s_2^*, s_3^*) with P_1^* = (0.5, 0.5), P_2^* = (0.5, 0.5) and P_3^* = (0.5, 0.5).

The matching pennies game also has two pure NEs (1, 0), (1, 0), (1, 0) and (0, 1), (0, 1), (0, 1), such that all players play heads with probability 1 or all players play tails with probability 1.

Example 2. Find mixed NEs in the following three-player game. Here, player 1 chooses between the rows *U* and *D*, player 2 chooses between the columns *L* and *R*, and player 3 chooses between the matrices *A* and *B*.

This example is cited Tayfun Sonnez's presentation [28].

	Player 3 chooses A			Player 3 chooses B	
	Player 2 chooses L	Player 2 chooses R		Player 2 chooses L	Player 2 chooses R
Player 1 chooses U	5, 5, 1	2, 1, 3	Player 1 chooses U	0, 2, 2	4, 4, 4
Player 1 chooses D	4, 7, 6	1, 8, 5	Player 1 chooses D	1, 1, 1	3, 7, 1

According to Tayfun's presentation, this game has only one pure NE (*U*, *R*, *B*); let us use the new algorithm to verify that this game has only a pure NE.

Player 1's payoff matrices are $A_{12} = \begin{pmatrix} 5, & 2 \\ 4, & 1 \end{pmatrix}$, $A_{13} = \begin{pmatrix} 0, & 4 \\ 1, & 3 \end{pmatrix}$

Player 2's payoff matrices are $A_{21} = \begin{pmatrix} 5, & 1 \\ 7, & 8 \end{pmatrix}^T$, $A_{23} = \begin{pmatrix} 2, & 4 \\ 1, & 7 \end{pmatrix}^T$

Player 3's payoff matrices are $A_{31} = \begin{pmatrix} 1, & 3 \\ 6, & 5 \end{pmatrix}^T$, $A_{32} = \begin{pmatrix} 2, & 4 \\ 1, & 1 \end{pmatrix}^T$

The same semantic rule M is used as defined in Example 1, such that $P_1 = (B_{11}, B_{21})$, $P_2 = (B_{12}, B_{22})$, $P_3 = (B_{13}, B_{23})$. where $B_{11} = B_{12} = B_{13}$ = (0, 1, 1); $B_{21} = B_{22} = B_{23}$ = (0, 0, 1) are TFNs defined in domain $[0,1] \times [0,1] \times [0,1]$.

By solving (7) for three players, one can obtain the following solution.

$$x = \frac{3}{5}, y = -\frac{2}{5}, z = 1$$

Since $y \notin [0, \ 1]$, this game does not have a mixed NE.

Even though different semantic rules are employed, one can obtain the same result. For example, if the sematic rules are used as follows:

$$P_1 = (B_{11}, B_{21}), P_2 = (B_{12}, B_{22}), P_3 = (B_{13}, B_{23}),$$

where $B_{11} = B_{12} = B_{13}$ = (0, 0, 1); $B_{21} = B_{22} = B_{23}$ = (0, 1, 1) are TFNs defined in domain $[0,1] \times [0,1] \times [0,1]$.

By solving (7) for three players, one can obtain the following solution.

$$x = \frac{2}{5}, y = \frac{7}{5}, z = 0$$

Since $y \notin [0, \ 1]$, this game does not have a mixed NE.

Example 3. Find mixed NEs for the following two-player game, which represents a wireless sensor network with the following bi-matrix.

		Player 2		
		Transmitting	Listening	Sleeping
Player 1	Transmitting	-4, -4	4, 4	-4, 1
	Listening	4, 4	1, 1	1, 2
	Sleeping	1, -4	2, 1	2, 2

DOI: 10.5963/IJCSAI0501001

$$A_{12} = \begin{pmatrix} -4 & 4 & -4 \\ 4 & 1 & 1 \\ 1 & 2 & 2 \end{pmatrix} \qquad A_{21} = \begin{pmatrix} -4 & 4 & 1 \\ 4 & 1 & 2 \\ -4 & 1 & 2 \end{pmatrix}^T$$

In this example, the number of strategies is three; we define m = 2, and divide the domain $U_i = [0,1](i=1,2)$ into two parts [0, 0.5] and [0.5, 1.0].

There are four combinations of domains for both players, as follows.

Domain1: $x \in (0, \ 0.5)$ and $y \in (0, \ 0.5)$; Domain2: $x \in (0, \ 0.5)$ and $y \in (0.5, \ 1.0)$

Domain3: $x \in (0.5, \ 1.0)$ and $y \in (0, \ 0.5)$; Domain4: $x \in (0.5, \ 1.0)$ and $y \in (0.5, \ 1.0)$

The semantic rule M is defined as follows.

$M_i : T_i(S_j) \to P_i (i=1,2; \ j=1,2,3)$, where $P_1 = (B_{11}, B_{12}, B_{13})$, $P_2 = (B_{21}, B_{22}, B_{23})$, $B_{i1} = (0,0,0.5)$, $B_{i2} = (0,0.5,1.0)$ and $B_{i3} = (0.5,1.0,1.0)(i=1,2)$.

At each domain, solve the following equations.

$$\begin{cases} \dfrac{\partial u_1(P_1, P_{-1})}{\partial P_1} = \dfrac{\partial}{\partial P_1}(P_1 \times A_{12} \times P_2^{\ T}) = 0 \\ \dfrac{\partial u_2(P_2, P_{-2})}{\partial P_2} = \dfrac{\partial}{\partial P_2}(P_2 \times A_{21} \times P_1^{\ T}) = 0 \end{cases} \tag{15}$$

Domain 1: $(0, \ 0.5) \times (0, \ 0.5)$; there is a solution $(x, y) = (4/11, 4/11)$ in this domain. The mixed NE (t, l, s) demonstrates the following probability distribution $P_1 = P_2 = (3/11, 8/11, 0)$ for both players.

Domain 2: $(0, \ 0.5) \times (0.5, \ 1)$; there is a solution $(x, y) = (3/8, 11/16)$ in this domain. The mixed NE (t, l, s) demonstrates the following probability distributions $P_1 = (1/4, 3/4, 0)$ and $P_2 = (0, 5/8, 3/8)$.

Domain 3: $(0.5, \ 1) \times (0, \ 0.5)$; there is a solution $(x, y) = (11/16, 3/8)$ in this domain. The mixed NE (t, l, s) demonstrates the probability distributions $P_1 = (0, 5/8, 3/8)$ and $P_2 = (1/4, 3/4, 0)$.

Domain 4: $(0.5, \ 1) \times (0.5, \ 1)$; (15) does not have a solution in this domain.

VI. CONCLUSIONS

This paper describes the derivation of the expected payoff function of polymatrix games, and presents the reasons why the expected payoff of each player in a polymatrix game is obtained as a sum of individual payoffs gained against each other player. Corresponding to the expected payoff function, a new algorithm to compute mixed NEs in polymatrix games is proposed. This paper proves that the new algorithm is able to compute NEs in polymatrix games within polynomial time.

Future work may be to compare the new algorithm with existing algorithms such as the extended Lemke-Howson method, the global Newton method, etc., and to extend the new algorithm to dynamic game theory.

ACKNOWLEDGMENT

The author would like to thank Dr. Tony Ponsford, Technical Director at Raytheon Canada to assist this research and proof reading. The author also appreciates anonymous reviewers' comments and suggestions.

REFERENCES

[1] C. E. Lemke and J. T. Howson, Jr, "Equilibrium Points in Bimatrix Games," *SIAM Journal on Applied Math*, vol. 12, pp. 413-423, 1964.

[2] J. Howson, "Equilibria of Polymatrix Games," *Management Science*, vol. 18, iss. 5, pp. 312-318, 1972.

[3] J. Rosenmuller, "On a generalization of the Lemke-Howson Algorithm to Noncooperative N-person Games," *SIAM Journal of Applied Mathematics*, vol. 1, pp. 73-79, 1971.

[4] R. Wilson, "Computing Equilibria of n-Person Games," *SIAM Journal of Applied Mathematics*, vol. 21, pp. 80-87, 1971.

[5] H. Scarf, "The Approximation of Fixed Points of a Continuous Mapping," *SIAM Journal on Applied Mathematics*, vol. 15, pp. 1328-1343, 1967.

[6] J. Dickhaut and T. Kaplan, "A Program for Finding Nash Equilibria," *The Mathematica Journal*, pp. 87-93, 1991.

DOI: 10.5963/IJCSAI0501001

[7] S. Govindan and R. Wilson, "A Global Newton method to Compute Nash Equilibria," *Journal of Economic Theory*, vol. 110, pp. 65-86, 2003.

[8] S. Govindan and R. Wilson, "Computing Nash Equilibria by Iterated Polymatrix Approximation," *Journal of Economic Dynamics and Control*, vol. 28, iss. 7, pp. 1229-1241, 2004.

[9] B. Blum, C. R. Shelton, and D. Koller, "A Continuation Method for Nash Equilibria in Structured Games," *Journal of Artificial Intelligence Research*, vol. 25, pp. 457-502, 2006.

[10] R. Porter, E. Nudelman, and Y. Shoham, "Simple Search Methods for Finding a Nash Equilibrium," *Games and Economic Behavior*, vol. 63, pp. 642-662, 2008.

[11] Y. Cai, O. Candogan, C. Daskalakis, and C. Papadimitriou, "Zero-Sum Polymatrix Games: A Generalization of Minmax," available at: people.csail.mit.edu/costis/zerosum_final3.pdf.

[12] S. Belhaiza, "On Perfect Nash Equilibria of Polymatrix Games," *Hindawi Publishing Corp. Game Theory*, vol. 2014, available at: http://dx.doi.org/10.1155/2014/937070.

[13] X. Chen and X. Deng, "3-Nash is PPAD-complete," ECCC, TR05-134, 2005.

[14] X. Chen and X. Deng, "Settling the Complexity of 2-Player Nash Equilibrium," ECCC, TR05-140, 2005.

[15] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou, "The Complexity of Computing a Nash Equilibrium," ECCC, TR05-115, 2005.

[16] C. Daskalakis and C. H. Papadimitriou, "Three-Player Games are Hard," ECCC, TR05-139, 2005.

[17] X. Chen, D. Paparas, and M. Yannakakis, "The Complexity of Non-Monotone Markets," In Proceeding of the 45th Annual ACM Symposium on Theory of Computing, pp. 181-190, 2013.

[18] A. Chakeri and F. Sheikholeslam, "Fuzzy Nash Equilibriums in Crisp and Fuzzy Games," *IEEE Transactions on Fuzzy Systems*, vol. 21, iss. 1, pp. 171-176, 2012.

[19] D. Garagic and J. B. Cruz, Jr, "An Approach to Fuzzy Noncooperative Nash Games," *Journal of Optimization Theory and Applications*, vol. 118, no. 3, pp. 475-491, 2003.

[20] S.H. Wu and V.W. Soo, "Fuzzy Game Theoretic Approach to Multi-Agent Coordination," T. Ishida: PRIMA'98, LNAI1599, pp. 76-87, 1999.

[21] A. Kaufmann, and M. M. Gupta, *Fuzzy Mathematical Models in Engineering and Management Science*, Elsevier, Amsterdam, 1998.

[22] L. Gao, "The Fuzzy Arithmetic Mean," *Fuzzy Sets and Systems*, vol. 107, pp. 335-348, 1999.

[23] L. A. Zadeh, "The Concept of Linguistic Variable and its Application to Approximate Reasoning-1," *Information Sciences*, vol. 8, pp. 199-249, 1975.

[24] H. J. Zimmermann, *Fuzzy Set Theory and Its Application*, 2nd ed., Kluwer Academic Publishers, Boston, 1991.

[25] L. Gao, "The Discussion of Applications of the Fuzzy Average to Matrix Game Theory," The Proceeding of CCECE2012, May 2012.

[26] L. Gao, "A Direct Algorithm for Computing Nash Equilibriums," *International Journal of Computer Science and Artificial Intelligence*, vol. 3, iss. 2, pp. 80-86, 2013.

[27] V. D. Blondel and J. N. Tsitsiklis, "A Survey of Computational Complexity Results," *Automatica, Elsevier*, vol. 36, pp. 1249-1274, 2000.

[28] T. Sonmez, available at: www.tayfunsonmez.net/wp-content/uploads/2013/10/E308SL4.pdf.

DOI: 10.5963/IJCSAI0501001