□ 研 究 論 文 □

韓國國防經營分析學會誌

제 31 권, 제 1 호, 2005. 6. 30.

# Blotto 게임을 풀기위한 새로운 근사해법 절차 (New Fictitious Play Procedure For Solving Blotto Games)

Jae-Yeong Lee,Moon-Gul Lee*

## Abstract

In this study, a new fictitious play (FP) procedure is presented to solve two-person zero-sum (TPZS) Blotto games. The FP solution procedure solves TPZS games by assuming that the two players take turns selecting optimal responses to the opponent's strategy observed so far. It is known that FP converges to an optimal solution, and it may be the only realistic approach to solve large games. The algorithm uses dynamic programming (DP) to solve FP subproblems. Efficiency is obtained by limiting the growth of the DP state space.

Blotto games are frequently used to solve simple missile defense problems. While it may be unlikely that the models presented in this paper can be used directly to solve realistic offense and defense problems, it is hoped that they will provide insight into the basic structure of optimal and near-optimal solutions to these important, large games, and provide a foundation for solution of more realistic, and more complex, problem

(**Keywords** : Fictitious Play, New FP Procedure, Two Person Zero Sum, Blotto Game Dynamic Programmi.)

* 국방대학교 관리대학원

# 1. Introduction

Fictitious play (FP), first introduced by Brown and Robinson (1951), is an iterative procedure used to approximate solutions to two-person zero-sum (TPZS) games. At each iteration of FP, each player chooses a pure strategy that is a best reply to the mixed strategy represented by the aggregation of all of other player's pure strategies played so far, assuming they will be chosen based on the empirical probability distribution induced by their historical frequency in all previous iterations. Fictitious play can be thought of as mimicking the behavior of players learning from their opponents.

The purpose of this thesis is to investigate the use of fictitious play in the solution of Blotto games and their generalizations. In Blotto games, opponents each allocate a limited number of forces to a specified number of areas. Payoffs in each area accrue to the players based on the number of forces assigned to each area. The main application of Blotto games has been the analysis of missile attack and defense problems.

This thesis is organized as follows: in Chapter 2, we introduce TPZS games, Blotto games and the solution procedure. In Chapter 3, we solve various versions of the problem using FP. Chapter 4 provides conclusions and suggests further work.

# 2. Two-person zero-sum game

## 2.1 Definitions

A two-person zero-sum (TPZS) game (von Neumann and Morgenstern, 1944) is a situation where there are two players having directly opposite interests. In a TPZS game, player 1 (also called X, the row player, or the maximizer) has m pure strategies and player 2 (Y, column player, minimizer) has n pure strategies. A player can commit to playing a pure strategy, or, by randomizing his choice among several pure strategies, he can employ a mixed strategy. A mixed strategy is represented by a vector of probabilities of choosing each pure strategy. For player 1, we write this vector as:

$$x = (x_1, \mathrm{K}\ , x_m)^T .$$

Because x is a vector of probabilities, we have the restrictions that

$$\sum_{i=1}^{m} x_i = 1$$

and

$$x_i \geq 0, \qquad i = 1,...,m \ .$$

Similar notation and restrictions are used for player 2, whose mixed strategy is written as:

$$y = (y_1, \mathrm{K}\ , y_n)^T \ .$$

In a TPZS game, each player chooses a strategy (pure or mixed), unknown to the other, and both strategies are revealed simultaneously. The result of the game depends on the strategy used by each player. If X and Y choose their ith and jth pure strategies, respectively, then the result of game, denoted aij, represents the amount that Y has to pay X. Equivalently, the payoffs to X and Y are aij and aij, respectively. Note that the sum of the two payoffs is zero, which explains the name of the game. If two players employ mixed strategies x and y(and a pure strategy is just a special case of a mixed strategy), then the payoff to player 1 is:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_i y_j a_{ij}$$

which can be seen as the expected payoff among all of the pure strategies represented by $x$ and $y$.

Therefore, aTPZS game is completely defined when the payoff for each pair of X and Y pure strategies is determined. These payoffs can be summarized in an mn matrix, generally referred to as a payoff matrix, i.e.

$$A = \begin{bmatrix} a_{11} & a_{12} & \mathrm{L} & a_{1n} \\ a_{21} & a_{22} & \mathrm{L} & a_{2n} \\ \mathrm{M} & \mathrm{M} & & \mathrm{M} \\ a_{m1} & a_{m2} & \mathrm{L} & a_{mn} \end{bmatrix},$$

and the payoff to player 1 is then $x^T Ay$.

In playing the game, both players are assumed to choose a strategy that achieves the most favorable outcome. This means that $X$ would choose the strategy that maximizes $x^T Ay$ over all choices of $y$. On the other hand, $Y$ would choose the strategy that minimizes $x^T Ay$ over all choices of $x$.

A TPZS game has an equilibrium point when each player can guarantee an optimal result by always choosing a single pure strategy. When equilibrium cannot be achieved, players must use mixed strategies to optimize the value of the game.

To choose the best randomized strategy, X must find the $x = (x_1, \mathrm{K}\ , x_m)$ to

$$\max_x\{\min_j[\sum_{i=1}^{m} x_i a_{ij}]: \sum_{i=1}^{m} x_i = 1,\ x_i \geq 0,\ i = 1,...,m\}$$

Similarly, *Y* must find the $y = (y_1, \mathrm{K}, y_n)$ to $\min_y\{\max_i[\sum_{j=1}^{n} a_{ij} y_j]: \sum_{j=1}^{n} y_j = 1,\ y_j \geq 0,\ j = 1,...,n\}$. Let $x^*$ and $y^*$ denote the optimal strategies for *X* and *Y*. Then, $v^* = (x^*)TAy^*$ is the *value* of the game. One of the central results of game theory states (Winston, 1991) that: $v^* = \max_x\{\min_j[\sum_{i=1}^{m} x_i a_{ij}]: \sum_{i=1}^{m} x_i = 1,\ x_i \geq 0,\ i = 1,...,m\}$

and $v^* = \min_y\{\max_i[\sum_{j=1}^{n} a_{ij} y_j]: \sum_{j=1}^{n} y_j = 1,\ y_j \geq 0,\ j = 1,...,n\}$.

## 2.2 Linear Programming

When the payoff matrix is specified and it is not too large, linear programming (Winston, 1991) can be used to find the optimal mixed strategies and the value of the game. For the maximizer (player 1), the problem is to find the mixed strategy $x = (x_1, \mathrm{K}, x_m)$ which maximizes $\min_j \sum_{i=1}^{n} x_i a_{ij}$. That is,

LP1: max $v$

subject to $\sum_{i=1}^{m} x_i a_{ij} - v \geq 0,\ j = 1,...,n$

$\sum_{i=1}^{m} x_i = 1,$ and $x_i = 1,...,m.$

Similarly, the minimizer (player 2) must solve LP 2:

LP2: min $w$

subject to $\sum_{j=1}^{n} y_j a_{ij} - w \leq 0,\ i = 1,...,m$

$\sum_{j=1}^{n} y_j = 1,$ and $y_j = 1,...,n.$

It is easy to show that problems LP1 and LP2 are duals of each other. Moreover, if (v*, x*) and (w*, y*) are optimal to problems LP1 and LP2, respectively, then v*=w*.

## 2.3 Fictitious play(FP): Brown-Robinson method

Fictitious play (FP)was introduced by Brown and Robinson (1951). It is an iterative solution procedure; in each iteration, players choose pure strategies that are the best response to the empirical mix of their opponents' pure strategies seen so far.

The FP procedure implemented here begins at iteration 1 with player 1 selecting that row maximizing the minimum row value, and player 2 selecting that column minimizing the maximum column value. Denote the players' pure strategies at iteration 1 as x(1) and y(1). These are vectors of all zeros except for a 1 at the selected row or column locations. At iteration 2, player 1 selects pure strategy x(2), which is the best row response to y(1) and player 2 selects pure strategy y(2), which is the best column response to x(1). And for general iteration k 2, player 1 selects the pure strategy x(k), which is the best row response to

$$\bar{y}^{(k-1)} = \frac{1}{k-1}\sum_{p=1}^{k-1} y^{(p)}, \text{ and}$$

player 2 selects pure strategy $y$(k), which is the best column response to

$$\bar{x}^{(k-1)} = \frac{1}{k-1}\sum_{p=1}^{k-1} x^{(p)}.$$

For computational purposes, $\bar{x}^{(k+1)}$ is conveniently updated from $\bar{x}^{(k)}$ and $x^{(k+1)}$ as follows:

$$\bar{x}^{(k+1)} = \left(\frac{k}{k+1}\right)\cdot\bar{x}^{(k)} + \left(\frac{1}{k+1}\right)\cdot x^{(k+1)}.$$

And similarly for player 2,

$$\bar{y}^{(k+1)} = \left(\frac{k}{k+1}\right)\cdot\bar{y}^{(k)} + \left(\frac{1}{k+1}\right)\cdot y^{(k+1)}.$$

Any limit points of the sequences $\{\bar{x}^{(k)}\}$ and $\{\bar{y}^{(k)}\}$ are optimal mixed strategy solutions to the game. Also upper and lower bounds on the value of the game, $v^*$, are determined at each game play. Specifically, at game iteration $k$,

$$\underline{v}_k \equiv (\bar{x}^{(k-1)})' A y^{(k)} \le v^* \le (x^{(k)})' A \bar{y}^{(k-1)} \equiv \bar{v}_k,$$

and both $\underline{v}_k$ and $\bar{v}_k$ converge to $v^*$, but not necessarily monotonically (Eagle and Washburn, 1991).

## 2.4 Blotto games

### 2.4.1 Definition

In a Blotto game, there are nl targets, or areas, for player 1 (the attacker) to attack and player 2 (the defender) to defend. The attacker chooses a vector of allocations x, where xk is the number of attacking units assigned to area k, and player 1 has f attackers to distribute, resulting in the constraint $\sum_{k=1}^{n} x_k \le f$. The defender chooses a vector of allocations y subject to $\sum_{k=1}^{n} y_k \le g$, where yk is the number

defenders assigned by player 2 to area k. The payoff is $\sum_{k=1} A(x_k, y_k)$ (See Washburn, 1994, pp.107-111 for a more complete discussion). All allocations are required to be nonnegative, and in a discrete Blotto game they are also required to be integers.

The number of pure strategies for player 1 is $\binom{n+f-1}{f}$ and, for player 2, $\binom{n+g-1}{g}$, both of which grow too fast to allow complete enumeration in even moderately sized games. Figure 1 displays a typical increase in the number of pure strategies for player 1 as f or n increase.

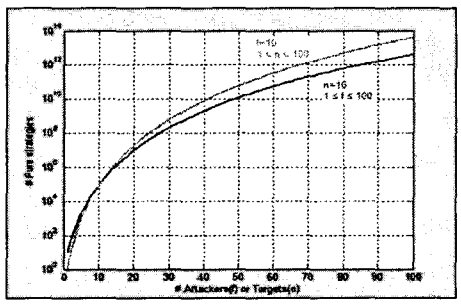


Figure 1. Number of pure strategies for player 1, for $f$=10 $1 \le n \le 100$, and for $n$=10

### 2.4.2 Playability

In Blotto games, it is sometimes convenient to represent a mixed strategy with the marginal distributions of the random vector

$X=(X_1,...,X_n)$, where $X_i$ is the random variable representing the number of attackers assigned to area $i$. Marginal distributions satisfying $\sum_k X_k = f$ are playable for the attacker. Similarly, the marginals for the random variable $Y=(Y_1,, Y_n)$ are playable for the defender if $\sum_k Y_k = g$. However, the typical approach is to relax these restrictions and simply require that $\sum_k E(X_k) = f$ and $\sum_k E(Y_k) = g$.

### 2.4.3 ILP Formulation of Blotto Games

Washburn (1994) presents the LP formulation of Blotto games using the marginal distributions. However, those formulations do not guarantee playability (although the discussions of those formulations claimthat playability is not an issue for large problems). We modify the formulation in Washburn (1994) by explicitly adding constraints that are sufficient to enforce playability.

The cost of such constraints is twofold: (1) they are sufficient, but not necessary conditions for playability, so the solutions we obtain are potentially suboptimal and (2) they introduce integer variables, rendering integer linear programming (ILP) formulations.

ILP 1 solves a Blotto defense game for the defender's marginals $(y_1,\mathrm{K},y_g)$, where yi represents the probability that jdefenders are used in any given area. Therefore, $\sum_t j \cdot y_i = E(Y_k)$.

$$\text{ILP1:} \quad \min_{y,c,d} \quad v = nc + df$$

$$s.t \quad \sum_{j=0}^{g} A(i,j) \cdot y_j \le c + d \cdot i; \quad i = 0,1...,f$$

$$\sum_{j=0}^{g} y_j \cdot j \le g/n$$

$$\sum_{j=0}^{g} y_j = 1$$

$$ycount_j = n \cdot y_j$$

$$y_j \ge 0, \ \text{for all } j = 0,1,...,g$$

$$ycount_j \in \{0,...,g\} \text{ and } d \ge 0$$

ILP 2 solves a Blotto attack game for the attacker's marginals $(x_1,\mathrm{K},x_f)$. The integer restrictions or *ycountj* and *xcounti*are used to require playability.

$$\text{ILP2:} \quad \max_{x,a,b} \quad v = na - bg$$

$$\text{s.t} \quad \sum_{i=1}^{f} A(i,j) \cdot x_i \ge a - b \cdot j; \quad j = 0,1...,g$$

$$\sum_{i=1}^{f} x_i \cdot i \le f/n$$

$$\sum_{i=1}^{f} x_i = 1$$

$$xcount_i = n \cdot x_i$$

$$x_i \ge 0, \ \text{for all } i = 0,1,...,f$$

$$xcount_i \in \{0,...,f\} \text{ and } b \ge 0$$

These playability constraints are too restrictive; they are sufficient to enforce playability but they are provably not necessary.

### 2.4.4 New Fictitious Play Procedure

We derive the dynamic programming recurrence relation for solving Blotto games with FP, using a general payoff function $A_k(x,y)$, which is the amount player 2 pays to player 1 when player 1 allocates x units to area k and player 2 allocates y units to area k. The total payoff is obtained by summing the rewards over the n areas. The recurrence can be solved without explicitly keeping track of every attack or defense played; rather, the information required is simply the number of times a given force

level (number of attackers or defenders) has been used in each area, over all attacks and defenses seen so far.

We first consider the defender's problem at FP iteration K, which is to allocate g defenders over n cities to minimize the expected payoff, given that K attacks have been observed so far. Each attack can be represented by a column vector $a^k = (a_1^k, a_2^k, \mathrm{K}, a_n^k)^T$, $k = 1, \mathrm{K}, K$. We define the values $r_i^j = \sum_k I_{(a_i^k = j)}$, where $I_{(a_i^k = j)}$ represents the indicator variable for the event, "the kth attack used j attackers in area i." Therefore, $r_i^j$ represents the number of times exactly j attackers have been used against area i. We first determine the value of placing g defenders optimally in area n. Then we define a recurrence relation on a value function $v_i(p)$ that represents the expected payoff of placing p defenders optimally in areas i, i+1,..., n. Then $v_1(g)$ is the solution to the original problem. The boundary condition is given by

$$v_n(q) = \frac{1}{K} \sum_{p=0}^{f} r_n^p A_n(p,q), \quad q = 0, \mathrm{K}, g, \qquad (1)$$

which is the total expected payoff when the defender uses $q$ defenders in area $n$. This states that the optimal defender strategy when only area $n$ remains is to allocate all $q$ remaining defenders to that area. The recurrence for $i \in \{1, \mathrm{K}, n-1\}$ is:

$$v_i(q) = \frac{1}{K} \min_{j=0,K,q} \left\{ \sum_{p=0}^{f} r_i^p A_i(p,j) + v_{i+1}(q-j) \right\} \qquad (2)$$

This is the expected payoff in area $i$ plus the expected payoff generated by placing the remaining $(q-j)$ defenders optimally in areas $i+1,, n$. The optimal defender allocation to area $i$ is the value of $j$ minimizing equation (2).

Similarly, for the attacker's problem, we assume that K defenses have been observed so far, where the kth defense is $d^k = (d_1^k, d_2^k, \mathrm{K}, d_n^k)^T$. The attacker wishes to allocate f forces over the n areas to maximize the expected payoff. Let $s_i^j = \sum_k I_{(d_i^k = j)}$ be the number of times j defenders are placed in area i. Define $w_i(p)$ as the maximum possible expected payoff in areas $i, \mathrm{K}, n$. The boundary conditions are:

$$w_n(p) = \frac{1}{K} \sum_{q=0}^{g} s_n^q A_n(p,q), \quad p = 0, \mathrm{K}, f, \qquad (3)$$

and the recurrence is given by:

$$w_i(p)=\frac{1}{K}\max_{j=0,K,p}\left\{\sum_{q=0}^{g}s_i^q A_i(j,q)+w_{i+1}(p-j)\right\}. \qquad (4)$$

The optimal solution for the attacker is represented by $w_1(f)$ and the corresponding decisions $j$ maximizing (4) for each area.

Blotto games can be extended immediately to the case where the attacker possesses different numbers of, say, two types of attacking units, f1 and f2, and the defender also has a supply of, say, two types of defending unit, g1 and g2. The payoff function now depends on the number of attackers and defenders of each type allocated to each area: $A_i(p_1,p_2,q_1,q_2)$. If we define $r_i^{j_1,j_2}$ as the number of times j1 attackers of type 1 and j2 of type 2 have been used in area i, and similarly for $s_i^{j_1,j_2}$, then our value functions are two-dimensional, with boundary conditions $v_n(q_1,q_2)=\frac{1}{K}\sum_{p_1=0}^{f_1}\sum_{p_2=0}^{f_2}r_n^{p_1,p_2}A_n(p_1,p_2,q_1,q_2)$,

(5)

and

$$w_n(p_1,p_2)=\frac{1}{K}\sum_{q_1=0}^{g_1}\sum_{q_2=0}^{g_2}s_n^{q_1,q_2}A_n(p_1,p_2,q_1,q_2), \qquad (6)$$

and recurrences

$$v_i(q_1,q_2)=\frac{1}{K}\min_{\substack{j_1=0,\cdots,q_1\\ j_2=0,\cdots,q_2}}\left\{\sum_{p_1=0}^{f_1}\sum_{p_2=0}^{f_2}r_i^{p_1,p_2}A_i(p_1,p_2,j_1,j_2)+v_{i+1}(q_1-j_1,q_2-j_2)\right\}. \qquad (7)$$

and

$$w_i(p_1,p_2)=\frac{1}{K}\max_{\substack{j_1=0,K,p_1\\ j_2=0,K,p_2}}\left\{\sum_{q_1=0}^{g_1}\sum_{q_2=0}^{g_2}s_i^{q_1,q_2}A_i(j_1,j_2,q_1,q_2)+w_{i+1}(p_1-j_1,p_2-j_2)\right\}. \qquad (8)$$

Clearly, the size of the static space grows with the product of the number of each type of attacker or defender. It is still manageable with just a few types of attacker or defender.

# 3. Data Analysis and Results

## 3.1 Model Implementation

The FP model is implemented in MATLAB (Version 6.5). The ILP solution procedure is implemented in GAMS (Revision 135, XA solver). Computations are done on a 1.5 GHz Intel Centrino-based laptop computer with 512 MB of RAM. All computer code appears in Appendices A and B.

## 3.2 Numerical Results

### 3.2.1 Rate of Convergence of FP

Define gap(k) to be the difference between the upper and lower bounds on the value of the game at FP iteration k. Consistent with earlier FP studies (Washburn, 2001), we find that the FP gap plotted against number of iterations is approximately asymptotically linear on a log-log plot. That is, for large enough k,

$$gap(k) \approx a/k^b, \text{ or}$$

$$\log(gap(k)) \approx \log(a) - b\log k,$$

where $k$ is number of iterations and $a$ and $b$ are fitted constants. Limited numerical experimentation suggests that using a least square fit and dropping first 100 iterations the intercept ($\log(a)$) increases with increasing $f$ or $g$, and the slope ($-b$) increases (to approximately -1/2) with increasing $n$. These observations are illustrated in Figures 2, 3 and 4.

| Case | f g n | # pure strategies | | Slope | Intercept | Final Gap (UB-LB) |
|---|---|---|---|---|---|---|
| | | Attacker | Defender | | | |
| A1 | 50, 200, 10 | 12565671261 | 1.76081E+15 | -0.48374 | 1.8887 | 3.8396 |
| A2 | 25, 100, 10 | 52451256 | 4.26342E+12 | -0.48378 | 1.5860 | 1.5260 |
| A3 | 5, 20, 10 | 2002 | 10015005 | -0.48376 | 0.8887 | 0.3052 |

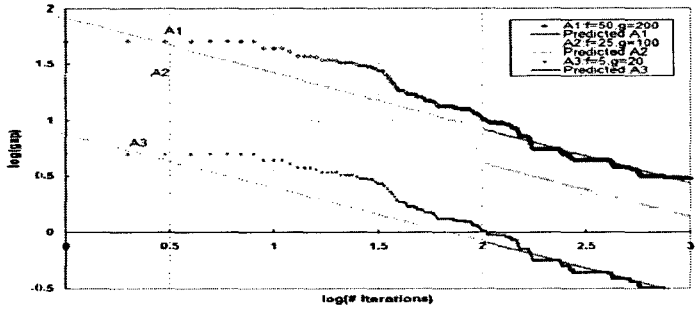


Figure 2. With $n$ fixed, the slope remains constant and the intercept increases with $f$ and $g$.

In Figure 2 we see the gap between the upper and lower bounds plotted against the number of iterations of fictitious play, on log-log scale. The slope of the fitted line (from the column labeled "slope" in the table above the plot) indicates the rate of convergence. Note that the slope is constant as f and g increase, and n remains fixed.

| Case | f, g, n | # pure strategies | | Slope (a) | Intercept (b) | Final Gap (UB-LB) | Standard Error |
|---|---|---|---|---|---|---|---|
| | | Attacker | Defender | | | | |
| B1 | 30, 50, 30 | 5.91E+16 | 3.33E+21 | -0.517 | 1.8541 | 2.0374 | 0.0094 |
| B2 | 30, 50, 20 | 1.89E+13 | 4.63E+16 | -0.543 | 1.8954 | 1.7824 | 0.0157 |
| B3 | 30, 50, 15 | 1.15E+11 | 4.79E+13 | -0.580 | 1.9463 | 1.5238 | 0.0187 |
| B4 | 30, 50, 10 | 2.12E+08 | 1.26E+10 | -0.588 | 1.8839 | 1.1999 | 0.0270 |
| B5 | 30, 50, 5 | 46376 | 316251 | -0.851 | 2.1746 | 0.4787 | 0.0469 |
| B6 | 30, 50, 2 | 31 | 51 | -1.093 | 1.4431 | 0.0216 | 0.0194 |

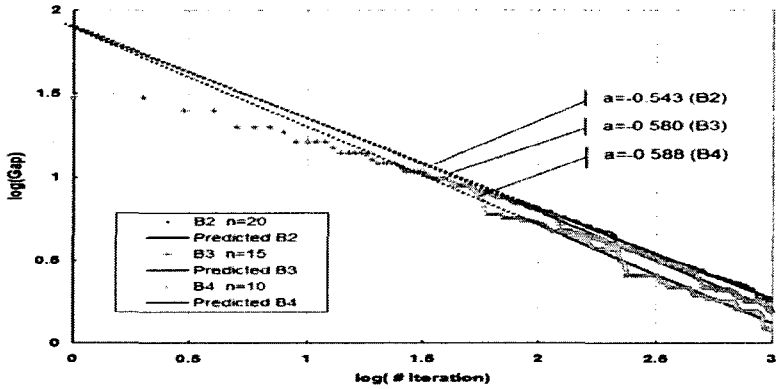


Figure 3. The best-fit slope increases with $n$ ($f$ and $g$ fixed).

If we increase n for a fixed f and g, we see in Figure 3 that the slope increases, and appears to approach a limit of -0.5 (Figure 4). This is consistent with conjectures of $1/\sqrt{k}$ convergence of FP.

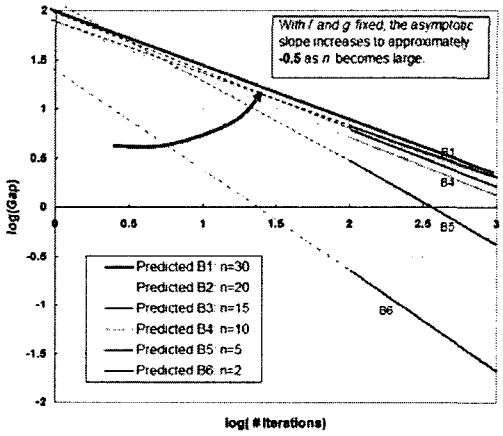


*Figure 4. Convergence of asymptotic slope*

### 3.2.2 Elapsed Time per FP Iteration

We observe that for all tested values of f, g and n, the elapsed time per FP iteration is constant as the number of FP iterations kincreases. This is illustrated in Figure 5 and occurs because the amount of

FP data required to be manipulated and stored does not increase with k.

| Iteration | 1000 | 2000 | 3000 | 4000 | 5000 | 6000 | 7000 | 8000 | 9000 | 10000 |
|---|---|---|---|---|---|---|---|---|---|---|
| f, g, n | Elapsed time (sec) | | | | | | | | | |
| 5, 6, 10 | 2.4 | 4.6 | 6.8 | 9.0 | 11.2 | 13.3 | 15.5 | 17.7 | 19.9 | 22.2 |
| 20, 25, 30 | 24.3 | 48.4 | 72.6 | 96.7 | 121.4 | 145.5 | 169.1 | 193.8 | 217.6 | 242.0 |
| 40, 45, 50 | 81.7 | 163.4 | 245.2 | 326.4 | 409.2 | 490.4 | 572.2 | 653.1 | 736.6 | 817.4 |

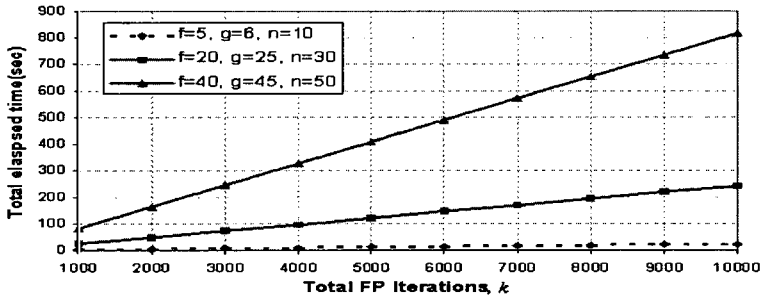


Figure 5. FP iterations vs. Elapsed time for 3 games

### 3.2.3 Comparisons with FP & ILP Procedure

Comparisons are made between the ILP and FP solution procedures. Three different payoff functions are examined.

#### 3.2.3.1 Convex Payoff Function

The convex payoff function is given by

$$A(x_i, y_i) = \max(x_i - y_i, 0).$$

Figure 6 shows how the two procedures performed.

| Run | f, g, n | FP (k = 2000) | | | | ILP | |
|---|---|---|---|---|---|---|---|
| | | Elapsed time | Upper | Lower | Gap | Elapsed time | Value of game |
| 1 | 3, 4, 5 | 1.962 | 2.208 | 2.179 | 0.029 | 0.10 | 2.2 |
| 2 | 6, 8, 5 | 2.774 | 4.415 | 4.358 | 0.057 | 0.18 | 4.4 |
| 3 | 12, 16, 5 | 4.657 | 8.831 | 8.717 | 0.114 | 0.22 | 8.8 |
| 4 | 15, 20, 5 | 5.668 | 11.039 | 10.896 | 0.143 | 0.19 | 11 |
| 5 | 30, 40, 5 | 11.147 | 22.077 | 21.792 | 0.285 | 0.25 | 22 |
| 6 | 60, 80, 5 | 24.535 | 44.154 | 43.584 | 0.570 | 0.12 | 44 |
| 7 | 120, 160, 5 | 59.766 | 88.308 | 87.168 | 1.140 | 0.21 | 88 |
| 8 | 150, 200, 5 | 83.801 | 110.385 | 108.960 | 1.426 | 0.12 | 111 |

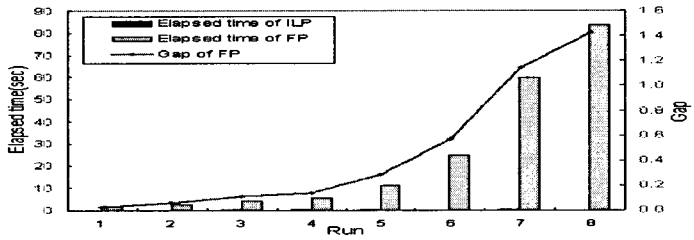


Figure 6. Comparison of FP and ILP procedures with convex payoff function

### 3.2.3.2 Capacitated Payoff Function

The capacitated payoff function is

$$A(x_i, y_i) = \begin{cases} \max(x_i - y_i, 0), & x_i - y_i \le cap \\ cap, & x_i - y_i > cap, \end{cases}$$

where *cap* is the maximum possible payoff. We note that the ILP objective function need not be either convex or concave Figure 7 shows how the two procedures performed.

| Run | $\mathcal{L}, g, n$ | FP ($k$ = 2000) | | | | ILP | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Elapsed time | Upper | Lower | Gap | Elapsed time | Upper | Lower | Gap |
| 1 | 3, 4, 5 | 1.743 | 2.208 | 2.179 | 0.029 | 0.180 | 2.200 | 2.200 | 0.000 |
| 2 | 6, 8, 5 | 2.704 | 2.958 | 2.872 | 0.086 | 0.100 | 3.120 | 2.800 | 0.320 |
| 3 | 12, 16, 5 | 4.697 | 3.935 | 3.767 | 0.168 | 0.100 | 4.114 | 3.800 | 0.314 |
| 4 | 15, 20, 5 | 5.748 | 4.252 | 4.078 | 0.174 | 0.170 | 4.333 | 4.000 | 0.333 |
| 5 | 30, 40, 5 | 11.226 | 4.884 | 4.619 | 0.265 | 0.190 | 5.143 | 4.200 | 0.943 |
| 6 | 60, 80, 5 | 24.536 | 5.350 | 4.960 | 0.389 | 1.420 | 5.807 | 4.200 | 1.607 |
| 7 | 120, 160, 5 | 59.946 | 5.622 | 5.130 | 0.492 | 62.380 | 6.230 | 4.200 | 2.050 |
| 8 | 150, 200, 5 | 82.278 | 5.713 | 5.144 | 0.570 | 102.170 | 6.338 | 4.091 | 2.247 |

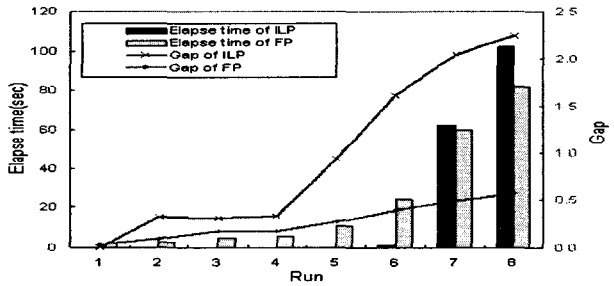


Figure 7. Comparison of FP of ILP procedures with the capacitated payoff function

### 3.2.3.3 Binary Payoff Function

The binary payoff function is

$$A(x_i, y_i) = \begin{cases} 0, & x_i - y_i \le 0 \\ 1, & x_i - y_i > 0 \end{cases}.$$

As with the capacitated payoff function, the ILP in this case need not be either convex or concave. Figure 8 shows how the two procedures performed.

| Run | f g n | FP (k = 2000) | | | | ILP | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Elapsed time | Upper | Lower | Gap | Elapsed time | Upper | Lower | Gap |
| 1 | 3, 4, 5 | 1.021 | 1.216 | 1.179 | 0.037 | 0.12 | 1.200 | 1.200 | 0.000 |
| 2 | 6, 8, 5 | 2.805 | 1.477 | 1.407 | 0.070 | 0.17 | 1.600 | 1.400 | 0.200 |
| 3 | 12, 16, 5 | 4.757 | 1.675 | 1.574 | 0.101 | 0.19 | 1.920 | 1.400 | 0.520 |
| 4 | 15, 20, 5 | 5.668 | 1.721 | 1.618 | 0.103 | 0.11 | 2.000 | 1.333 | 0.667 |
| 5 | 30, 40, 5 | 11.186 | 1.850 | 1.694 | 0.156 | 0.30 | 2.000 | 1.400 | 0.600 |
| 6 | 60, 80, 5 | 24.756 | 1.921 | 1.716 | 0.205 | 2.57 | 2.167 | 1.400 | 0.767 |
| 7 | 120, 160, 5 | 60.758 | 1.990 | 1.703 | 0.287 | 61.96 | 2.182 | 1.400 | 0.782 |
| 8 | 150, 200, 5 | 83.260 | 2.012 | 1.686 | 0.326 | 149.89 | 2.214 | 1.400 | 0.814 |

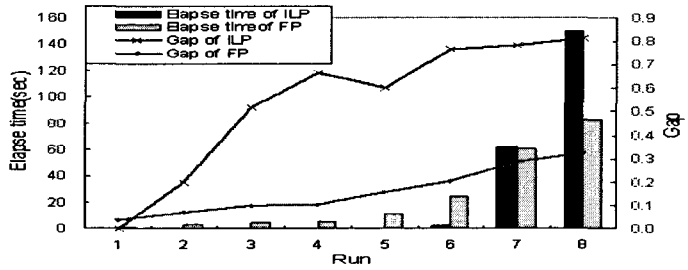


Figure 8. Comparison of FP and ILP procedures with the binary payoff function

### 3.3 Conclusions

As has been observed in earlier FP studies the FP gap (the difference between the upper and lower bounds or game value) as a function of number of FP iterations, k, is approximately

$$gap(k) \approx a/k^b,$$

for large enough k.

The best-fit $a$ increases with $f$ and $g$, and the best-fit $b$ decreases to approximately 1/2 with increasing $n$.

Because of efficiencies realized in the DP procedure used to solve the FP subproblems, the computation time required for each FP iteration is approximately constant as the number of FP iterations increases, for fixed $f$, $g$ and $n$.

For the convex payoff function tested, the ILP formulation solved with GAMS was faster and more accurate than the FP procedure.

For the non-convex payoff functions tested, the FP procedure was more competitive and sometimes significantly outperformed than ILP procedure.

## 4. Conclusions and Further Study

We propose a new efficient fictitious play (FP) procedure to solve two-person zero-sum Blotto games. The algorithm uses dynamic programming (DP) to solve the FP subproblems at each iteration. By representing intermediate mixed strategies through marginal distributions are keep the state space of the DP manageable and independent of the number of iterations. Although our experiments considered

one type of attacker and one type of defender, we indicate how to generalize this procedure to cases with more than one type of attacker or defender (or both).

During this study, we identified other topics for further investigations. The first is to investigate generalizations of Blotto games in which defenders can be placed in such a way as to defend multiple areas at once. This is closer to the real situation with missile defense. The second is to explore the issue of playability in the ILP formulations. Our proposed playability constraint is currently too restrictive. we have provided examples in which the optimal solution to the ILP, with the playability constraint, is not equal to the value of the game. Further research should explore less restrictive, alternate formulations of playability constraints. It is possible (although unlikely) that less restrictive playability constraints would also yield more efficiently solvable ILP, making that approach competitive with the DP-based procedure for larger problems.

## References


[1] Alan R. Washburn, 1994. Two-Person Zero-Sum Games, Institute for Operations Research and the Management Sciences. pp. 36-38

[2] Alan Washburn, 2001, A New kind of fictitious play, Naval Postgraduate School, http://diana.or.nps.mil/~washburn/ModFicPlay/ pp. 1-2

[3] Brown, G.W., 1951, "Iterative Solutions of Games by Fictitious Play." In T.C. Koopmans (ed.), Activity Analysis of Production and Allocation, Cowles Commission Monograph 13, pp. 377-380

[4] Eagle, J. N. and Washburn, A.R. 1991. Cumulative Search-Evasion Games. Naval Research Logistics, Vol. 38, pp.495-510

[5] Robinson, J., 1951. An Iterative Method of Solving a Game. Annals of Mathematics, 54, pp. 296-301

[6] von Neumann, J., and O. Morgenstern. 1944. Theory of Games and Economic Behavior. Princeton U. Press

[7] Winston, W. I., 1991. Operations Research Applications and Algorithms, 2nd ed., PWSKENT, Co.


There are many methods in the field of decision analysis to help a decision maker