TOTAL AND PARTIAL CONFLICT GAME ANALYSIS USING MAPLE

William P. Fox
Naval Postgraduate School

Abstract

We apply principles of optimization, linear programming and nonlinear programming, to find the solutions for Nash equilibriums and Nash arbitration in two person total conflict and partial conflict games. Linear programming can always be used to find the solution to the zero-sum and constant sum games, linear programming may also be used in a two-Person, two-strategy game where no pure strategy exists by adding an equaling constraint. For games with more than two strategies for a player, we recommend a nonlinear approach to finding the Nash equilibriums for pure and mixed strategies. For prudential strategies, we have shown how linear programs can be used for each player to find their security levels. We then allow for communications and apply the Nash arbitration methodology as a nonlinear optimization problem. We illustrate all these using MAPLE procedures.

Key words: Total conflict games, partial conflict games, zero sum, nonzero sum games, game theory, Nash equilibrium, prudential strategies, security levels, Nash arbitration, linear programming, nonlinear programming, MAPLE

Introduction

In our interdisciplinary Department of Defense Analysis at the Naval Postgraduate School, we teach a three course sequence in mathematical modeling for decision making. In the first course, we usually teach basic linear programming both using the two-variable graphical simplex technique and the Excel Solver using SimplexLP. We have also added the computer algebra system, MAPLE for advanced student work.

In this 3rd course, we teach the basic concepts and solution techniques for game theory. In our class we use the Straffin text [8] as well as Chapter 10 from Giordano, Fox, and Horton [4]. We do not cover the basic solution techniques in this paper other than to illustrate the movement diagram. We also illustrate the use of MAPLE in solving game theory.

Our students must complete a course project of their own choice using one of the modeling techniques from class. Students use the modeling process in their project: they identify the problem; they list the appropriate assumptions with justifications; they explain why their modeling technique is selected; they solve the model; interpret the solution; perform sensitivity analysis (if applicable); and they discuss strengths and weaknesses of their modeling approach. Here is short list of some of the game theory projects:

- Game Theory with US and Non-State actors.
- Game Theory in Cameroon-Nigeria dispute.
- Game Theory in PMI and US military tasks.
- US-China Game.
- The Somali Pirates game.
- US-Afghanistan drug dilemma.
- US-Afghanistan Regional Game.
- US Coin Operations Game.
- Dealing with Safe Havens as a Game.
- IEDS and Counter-IEDS as a Game.
- Game theory for Courses of Combat Actions
- Game Theory and Dark Money Networks

In the past, our coverage did not cover much linear programming or nonlinear programming,

so our solution processes were limited to two-person, two strategy games using the algebraic method or other short cut methods because of the complexity of the solution mechanics. Recently, we have added more applications of linear programming and a non-linear template as a solution technique so students might add more reality to the number of possible strategies available to the players.

Let's define a generic simultaneous two person game theory payoff matrix as shown in Table 1.

Total Conflict Games

In our total conflict games we make a few assumptions about the game. We assume the following: our players are rational in that each player wants to achieve the best solution possible versus their opponent; the games are simultaneous; the games are repetitive, and each player has perfect knowledge concerning their opponent. Further there are two rules that can make a game a total conflict game. They are that either $M_{ij} + N_{ij} = 0$ or $M_{ij} + N_{ij} = C$ where C is a constant, for all i and j. We define the linear programming formulation that solves all total conflict games obtaining either the pure strategy or mixed strategy optimal solution.

Our knowledge of the zero-sum game and the Primal-Dual relations suggest formulating Rose's game and find Colin's solution through the dual solution. This works well if you have a single problem to solve. But what if you have many games to consider and you only have Maple. How best to construct a technology assistant?

Treating the zero-sum game as above translates into two linear programming formulations, one for each maximizing player. This is easily handled inside a Maple procedure. Further if any pairs ($M_{m,n}$, $N_{m,n}$) contain a negative entry then there is a chance that the

Rose's Strategies			Colin's Strategies	
	Column 1	*Column 2*	…	*Column n*
Row 1	$M_{1,1}$,$N_{1,1}$	$M_{1,2}N_{1,2}$	…	$M_{1,n}N_{1,n}$
Row 2	$M_{2,1}N_{2,1}$	$M_{2,2}$,$N_{2,2}$	…	$M_{2,n}$,$N_{2,n}$
. . .	. . .	. . .	. . .	. . .
Row m	$M_{m,1}$,$N_{m,1}$	$M_{m,2}N_{m,2}$	…	$M_{m,n}$,$N_{m,n}$

Table 1. General payoff matrix of a two-person total conflict game with outcomes (M_{ij},N_{ij}) when players, play strategies i and j respectively.

game solution can be negative. Since the game solution will be a decision variable in our formulation, we must account for that possibility. Our best recommendation is to use the method suggested by Winston [23, p.172-178] to replace any variable that could take on negative values with the difference in two positive variables, $x_j - x'_j$. We assume that the value of the game could be positive or negative. The other values we are looking for are probabilities that are always between 0 and 1. Since this occurs only in the value of the game, we use as a substitution of variables, such as $V=p_1-p_2$ or $v=q3-q4$ as in equations *(1)* and *(2).*

Maximize p_1-p_2

Subject to:

$$
\begin{aligned}
&M_{1,1}x_1 + M_{2,1}x_2 + ... + M_{m,1}x_n - p_1 + p_2 \geq 0 \\
&M_{1,2}x_1 + M_{2,2}x_2 + ... + M_{m,2,}x_n - p_1 + p_2 \geq 0 \\
&... \\
&M_{1,m}x_1 + M_{2,m}x_2 + ... + M_{m,n}x_n - p_1 + p_2 \geq 0 \\
&x_1 + x_2 + ... + x_n = 1 \\
&\textit{Nonnegativity}
\end{aligned}
\qquad (1)
$$

where the weights x_i yields Rose strategy and the value of V is the value of the game to Rose.

Maximize q_3-q_4

Subject to:

$$
\begin{aligned}
&N_{1,1}y_1 + N_{1,2}y_2 + ... + N_{1,m}y_n - q_3 + q_4 \geq 0 \\
&N_{2,1}y_1 + N_{2,2}y_2 + ... + N_{2,m}y_n - q_3 + q_4 \geq 0 \\
&... \\
&N_{m,1}y_1 + N_{m,2}y_2 + ... + N_{m,n}y_n - q_3 + q_4 \geq 0 \\
&y_1 + y_2 + ... + y_n = 1 \\
&\textit{Nonnegativity}
\end{aligned}
\qquad (2)
$$

where the weights y_i yield Colin's strategy and the value of v_3-v_4 is the value of the game to Colin.

We wrote a small procedure in Maple to solve for the solution when the user evokes the TotalConflctGame procedure and enters the number of strategies for each player followed by the payoff matrix as we will illustrate.

```
TotalConflctGame := proc( r, c, A)

    with(LinearAlgebra) : with(Optimization) :
    #make a local copy of A that we can change
    M := Matrix( A ) : B1 := Transpose(A) : B := -A :
    X := `<,>`(seq(x[i], i = 1 ..r));
    Y := `<,>`(seq(y[i], i = 1 ..c));

    Cnst1 := {seq((B.Y)[i] ≥ p1 − p2, i = 1 .. r) , add(y[i], i = 1 .. c)
    = 1};
    Cnst := { seq((B1.X)[i] ≥ q1 − q2, i = 1 .. c), add(x[i], i = 1 .. r)
    = 1};
     Colin := LPSolve(p1 − p2, Cnst1, assume = nonnegative,
    maximize);
     Rose := LPSolve(q1 − q2, Cnst, assume = nonnegative,
    maximize);
    print(Rose, Colin);
end proc;
```

Example 1. Rose versus Colin

		Colin	
		C1	C2
Rose	R1	2	-3
	R2	0	2

The solution is shown using Maple for a game that easily can be solved by hand with short-cut methods.

First we enter the payoff matrix,

```
> A1 := Matrix([[2,-3],[0,2]]);
```

$$A1 := \begin{bmatrix} 2 & -3 \\ 0 & 2 \end{bmatrix}$$

We note there are 2 strategies each. So we evoke our procedure

```
> TotalConflictGame(2,2,A1);
```

$$[0.571428571723600, [q1 = 0.571428571723600, q2 = 0., x_1 = 0.285714285861800, x_2 = 0.714285714138200]], [-0.571428569363376, [p1 = 0., p2 = 0.571428569363376, y_1 = 0.714285714285714, y_2 = 0.285714285714286]]$$

Our answers are printed by the procedure as the output with Rose's followed by Colin's.

The value of the game to our player's is (0.571428571723600, -0. 571428571723600) when Rose play's *0.28571428586 R1, 0.714285714138200 R2* while Colin plays *0.714285714138200 C1, 0.714285714138200 C2*.

Example 2. Rose versus Colin.

Consider a larger game where Rose has three strategies and Colin has six strategies. The payoff matrix will be as shown in matrix, A.

```
>
A := Matrix([[4,-4,3,2,-3,3],[-1,-1,-2,0,0,4],[-1,2,1,-1,2,-3]]);
```

$$A := \begin{bmatrix} 4 & -4 & 3 & 2 & -3 & 3 \\ -1 & -1 & -2 & 0 & 0 & 4 \\ -1 & 2 & 1 & -1 & 2 & -3 \end{bmatrix}$$

We evoke the procedure as follows and obtain our outputs.

```
> TotalConflictGame(3,6,A);
```

$$[-0.0714285700799631, [q1 = 0., q2 = 0.0714285700799631, x_1 = 0.238095238267338, x_2 = 0.214285714156640, x_3 = 0.547619047576023]], [0.0714285720985307, [p1 = 0.0714285720985307, p2 = 0., y_1 = 0., y_2 = 0.357142857050661, y_3 = 0., y_4 = 0.571428571711307, y_5 = 0., y_6 = 0.0714285712380325]]$$

The solution provides the values of the game to Rose and Colin as (-0.07142857, 0.07142857). To obtain this optimal solution our players must play their strategies with the following mixed solution rounded as Rose plays, *0.23809* R_1, *0.2142857* R_2, and *0.547619* R_3 and Colin never plays C1 C3, or C5 but plays the following mix of C2, C4, and C6 of 0.357142857 C_2, 0.57142857 C_4, and 0.07142857 C_6.

Although we did not illustrate it here, linear programming will find the optimal solutions when the solution is pure strategy or mixed strategy. One limitation to our Maple procedure is finding alternate optimal solutions. We suggest one method if a pure strategy solution exists is to try the Saddle-Point method to look for possible alternate optimal solutions.

Partial Conflict Games

Let's first define a partial conflict game. As opposed to a total conflict game where if a player wins *x* his opponent loses *x*, in a partial conflict game the players are not strictly opposed, so it is possible for both players to win or lose some value.

In a partial sum game the sum of the values for the two players do not sum to zero. For

example, consider the following game where the sums of the outcomes do not all sum to zero.

		Player II	
	C_1		C_2
R_1	(2, 4)		(1, 0)
Player I			
R_2	(3, 1)		(0, 4)

In Figure 1 we note that a plot of the payoffs to each player do not lie on a line, indicating that the game is a partial conflict game because total conflict game values do lie in a straight line.

What are the objectives of the players in a partial conflict game? In total conflict, each player attempts to maximize his payoffs and necessarily minimizes the other player in the process. But in a partial conflict game, a player may have any of the following four objectives taken from Giordano et al. [6]

1. **Maximize his payoffs.** Each player chooses a strategy in an attempt to maximize his payoff. While he reasons what the other player's response will be he does not have the objective of insuring the other player gets a "fair" outcome. Instead, he "selfishly" maximizes his payoff [Giordano, 6].

2. **Find a stable outcome.** Quite often players have an interest in finding a stable outcome. *A Nash equilibrium outcome is an outcome from which neither player can unilaterally improve*, and therefore represents a stable situation. For example, we may be interested in determining whether two species in a habitat will find equilibrium and coexist, or will one species dominate and drive the other to extinction? [Giordano, 6] The Nash equilibrium is named in honor of John Nash [7] who proved that every two-person game has at least one equilibrium in either pure strategies or mixed strategies.

3. **Minimize the opposing player.** Suppose we have two corporations whose marketing of products interact with each other, but not in total conflict. Each may begin with the objective of maximizing his payoffs. But, if dissatisfied with the outcome, one, or both corporations, may turn hostile and choose the objective of minimizing the other player. That is, a player may forego their long-term goal of maximizing their own profits and choose the short term goal of minimizing the opposing player's profits. For example, consider a large, successful corporation

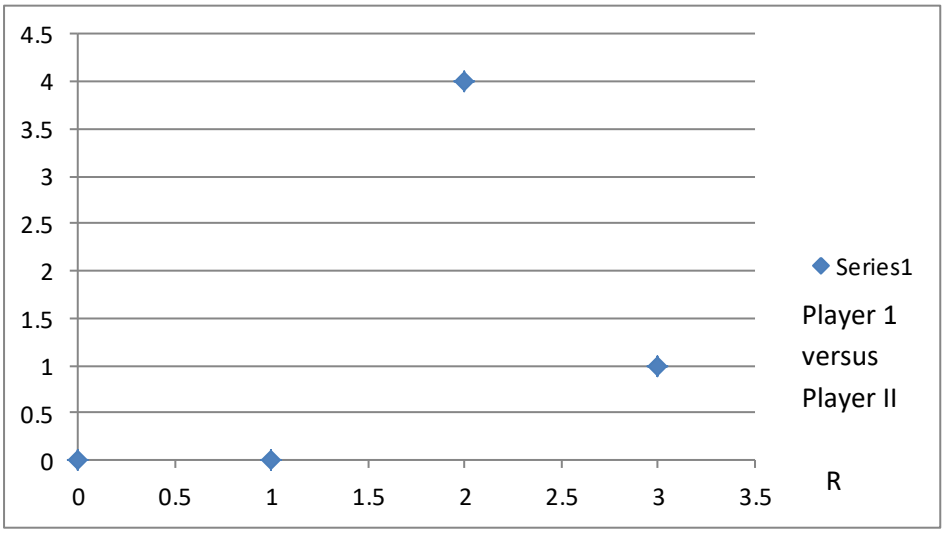

Figure 1. Payoffs in a partial conflict game do not lie on a line.

attempting to bankrupt a "start-up venture" in order to drive him out of business, or perhaps motivate him to agree to an arbitrated "fair" solution. [Giordano, 6]

4. **Find a "mutually fair" outcome, perhaps with the aid of an arbiter.** Both players may be dissatisfied with the current situation. Perhaps, both have a poor outcome as a result of minimizing each other. Or perhaps one has executed a "threat' as we study below, causing both players to suffer [Giordano, 6]. In such cases the players may agree to abide by the decision of an arbiter who must then determine a "fair" solution and does according to Nash [7].

In this introduction to partial conflict games, we will assume that both players have the objective of maximizing their payoffs. Next we must determine if the game is played without communication or with communication. "Without communication" indicates that the players must choose their strategies without knowing the choice of the opposing player. For example, perhaps they choose their strategies simultaneously. The term "with communication" indicates that perhaps one player can move first and make his move known to the other player, or that the players can talk to one another before they move. We assume that our games do not allow communication and are played simultaneously.

Further we assume our players are rational, attempting to obtain their best outcomes, players have perfect knowledge, and that games are repetitive.

One method to find a pure strategy solution is the movement diagram. We define the movement diagram as follows:

***Movement Diagram*:** For Player one, examine the first value in the coordinate and compare R_1 to R_2. For each C_1 and C_2 draw an arrow from the smaller to larger values between R_1 and R_2. For Player two examine the second value in the coordinate and compare C_1 to C_2. For each R_1 and R_2 draw an arrow from the smaller to larger values between C_1 and C_2.

Example 3. Player I versus Player II

For example, under C_1, we draw an arrow from 2 to 3 and under C_2 from 0 to 1. Under R_1 we draw the arrow from 0 to 4 and under R_2 from 1 to 4. We show this in Figure 2.

		Player II	
	C_1		C_2
R_1	(2, 4)	←	(1, 0)
Player I	↓		↑
R_2	(3, 1)	→	(0, 4)

Figure 2. Movement diagram.

Figure 2, the arrows indicate "false" in all directions so there is no pure strategy.

We follow the arrows. If the arrows lead us to a value or values where no arrows points out then we have a pure strategy solution. If the arrows move in a clockwise or counter-clock wise direction then we have no pure strategy solution. Here we move counter-clock wise and have no pure strategy solution. As Nash proved all games have a solution either by pure or mixed strategies. As a matter of fact others (Barron [1]; Houseman and Gillman[5]) have shown that some partial conflict games have both a pure and mixed (equalizing) strategy.

We begin by defining the mixed (equalizing) strategy for a partial conflict game.

Rose's game: Rose maximizing, Colin "equalizing" is a total conflict game that yields Colin's equalizing strategy.

Colin's game: Colin maximizing, Rose "equalizing" is a total conflict game that yields Rose's equalizing strategy.

Note: If either side plays its equalizing strategy, then the other side "unilaterally" cannot improve its own situation (it stymies the other player).

We will call this strategy, an equalizing strategy. Each player is restricting what his opponent can obtain by insuring no matter what they do that his opponent always gets the identical solution (Straffin [8]).

Methods to Obtain the Equalizing Strategies

We present two methods to obtain equalizing strategies and we will apply these methods to our previous example. The two methods are: linear programming and nonlinear programming. We state here that linear programming works well here only because each player has only two strategies. It gets very complicated if a player has three strategies because we do not know a priori which strategies will be equalized and which ones might not be played at all.

Linear Programming with Two Players and Two Strategies Each

This translates into two maximizing linear programming formulations as shown in Equations (3) and (4). Formulation (3) provides the Nash equalizing solution for Colin with strategies played by Rose while formulation (4) provides the Nash equalizing solution for Rose and strategies played by Colin. The two constraints representing strategies are implicitly equal to each other per this formulation (Fox, [3]).

Maximize V
Subject to:

$$N_{1,1}x_1 + N_{2,1}x_2 - V \geq 0$$
$$N_{1,2}x_1 + N_{2,2}x_2 - V \geq 0$$
$$(N_{1,1} - N_{1,2})x_1 + (N_{2,1} - N_{2,2})x_2 = 0 \qquad (3)$$
$$x_1 + x_2 = 1$$
Nonnegativity

Maximize v
Subject to:

$$M_{1,1}y_1 + M_{1,2}y_2 - v \geq 0$$
$$M_{2,1}y_1 + M_{2,2}y_2 - v \geq 0$$
$$(M_{1,1} - M_{2,1})y_1 + (M_{1,2} - M_{2,2})y_2 = 0 \qquad (4)$$
$$y_1 + y_2 = 1$$
Nonnegativity

With our example, we obtain the following formulation

Maximize V
Subject to:

$$4x_1 + x_2 - V \geq 0$$
$$0x_1 + 4x_2 - V \geq 0$$
$$4x_1 - 3x_2 = 0$$
$$x_1 + x_2 = 1$$
Nonnegativity

and

Maximize v
Subject to:

$$2y_1 + y_2 - v \geq 0$$
$$3y_1 + 0y_2 - v \geq 0$$
$$-y_1 + y_2 = 0$$
$$y_1 + y_2 = 1$$
Nonnegativity

The solution, constructed by adding the equalizing constraint and solving in Maple is *3/7 x_1, 4/7 x_2* corresponding to *3/7 R_1, 4/7 R_2 and ½ y_1, ½ y_2* corresponding to *½ C_1, ½ C_2*. The Nash equilibrium is (3/2, 16/7).

Nonlinear Programming Approach for Two or More Strategies for Each Player

For games with two players and more than two strategies each, we present the nonlinear optimization approach by Barron [1]. Consider a two person game with a payoff matrix as before. Let's separate the payoff matrix into two matrices **M** and **N** for players I and II. We solve the following nonlinear optimization formulation in expanded form, in Equation (5).

$$Maximize \sum_{i=1}^{n}\sum_{j=1}^{m} x_i a_{ij} y_j + \sum_{i=1}^{n}\sum_{j=1}^{m} x_i b_{ij} y_j + - p - q$$

Subject to

$$\begin{gathered}\sum_{j=1}^{m} a_{ij} y_j \le p, \quad i = 1,2,\dots,n, \\ \sum_{i=1}^{n} x_i b_{ij} \le q, \quad j = 1,2,\dots,m, \\ \sum_{i=1}^{n} x_i = \sum_{j=1}^{m} y_j = 1 \\ x_i \ge 0, y_j \ge 0\end{gathered} \qquad (5)$$

Our procedure in Maple is as follows:

```
PartialConflictGame := proc( ar, ac, A, B, ip, iq)

  with(LinearAlgebra) : with(Optimization) :

  X := `<,>`(seq(x[i], i = 1 ..ar));
  Y := `<,>`(seq(y[i], i = 1 ..ac));
  Cnst := {seq((A.Y)[i] ≤ p, i = 1 .. ar), seq((Transpose(X).B)[i]
   ≤ q, i = 1 .. ac), add(x[i], i = 1 .. ar) = 1, add(y[i], i = 1 .. ac) = 1};
  objective := expand(Transpose(X).A.Y + Transpose(X).B.Y-p
   -q);
  QPSolve(objective, Cnst, assume = nonnegative, maximize,
   initialpoint = {p = ip, q = iq});
```

We note that our NLP method is quadratic programming so we can later change the initial starting points *(ip, iq)* to look for other solutions.

We return to our previous example. We define *M* and *N* as:

$$\boldsymbol{M = \begin{bmatrix} 2 & 1 \\ 3 & 0 \end{bmatrix} \; and \; N = \begin{bmatrix} 4 & 0 \\ 1 & 4 \end{bmatrix}}$$

We define x_1, x_2, y_1, y_2 as the probabilities for players playing their respective strategies.

By substitution and simplification, we obtain the formulation as

Maximize

$6y_1x_1 + 4y_1x_2 + x_1y_2 + 4x_2y_2 - p - q$

Subject to:

$x_1+x_2=1$
$y_1+y_2=1$
$4x_2-q\le 0$
$4x_1+x_2-q\le 0$
$2y_1+y_2-p\le 0$
$3y_1-p\le 0$
Nonnegativity

In Maple, we can enter our payoff matrices and evoke the PartialConflictGame procedure.

```
> A := Matrix([[2, 1], [3, 0]]);
```

$$A := \begin{bmatrix} 2 & 1 \\ 3 & 0 \end{bmatrix}$$

```
> B := Matrix([[4, 0], [1, 4]]);
```

$$B := \begin{bmatrix} 4 & 0 \\ 1 & 4 \end{bmatrix}$$

```
> ar := 2 : ac := 2;
```

$$ac := 2$$

```
> PartialConflictGame(ar, ac, A, B, 10, 10);
```

$$[0., [p = 1.50000000000000, q = 2.28571428571429, x_1 = 0.428571428571429, x_2 = 0.571428571428571, y_1 = 0.500000000000000, y_2 = 0.500000000000000]]$$

We found the exact same solution as before as expected.

Finding a Solution

According to Straffin [8], a Nash equilibrium is a solution if and only if it is unique and Pareto Optimal. Pareto optimal refers to the northeast region of a payoff polygon where the payoff polygon is found as the convex set formed by the outcome coordinates, Figure 3.

We see in the figure that the Nash equilibrium (1.5, 2.28) is not Pareto optimal and not the solution that we should seek.

At this point, we might try to allow communication and try strategic moves which we do not describe here but can be reviewed in Giordano, et al. [6] or Straffin [8]. Next, we consider moving on to arbitration.

Prudential Strategies

We define a prudential strategy as the strategies achieved by each player as they attempt to optimize their own payoffs. Rose seeks to determine her optimal set of strategies to optimize her payoff while Colin determines his strategy to optimize his payoff. We might treat these as linear programs for each player since both players desire to optimize their outcomes. Straffin [8] refers to values of these two games, as the security levels for the players. These can be used later in the Nash arbitration method.

```
PrudentialStrategies := proc( ar, ac, A, B)
    with(LinearAlgebra) : with(Optimization) :
   X := `<,>`(seq(x[i], i = 1 ..ar));
   Y := `<,>`(seq(y[i], i = 1 ..ac));
   CnstR := {seq((Transpose(X).A)[i] ≥ p1 − p2, i = 1 .. ar),
    add(x[i], i = 1 .. ar) = 1};
   CnstC := {seq((B.Y)[i] ≥ q1 − q2, i = 1 .. ac), add(y[i], i = 1
   .. ac) = 1};
    Obj1 := p1 − p2; Obj2 := q1 − q2;
  SLR := LPSolve(Obj1, CnstR, assume = nonnegative, maximize) :
    SLC := LPSolve(Obj2, CnstC, assume = nonnegative, maximize) :
  print(SLR, SLC);
end proc;
```

```
> A := Matrix([[2, 1], [3, 0]]);
```

$$A := \begin{bmatrix} 2 & 1 \\ 3 & 0 \end{bmatrix}$$

```
> B := Matrix([[4, 0], [1, 4]]);
```

$$B := \begin{bmatrix} 4 & 0 \\ 1 & 4 \end{bmatrix}$$

```
> ar := 2 : ac := 2;
```

```
> PrudentialStrategies(ar, ac, A, B);
```

$$[1., [p1 = 1., p2 = 0., x_1 = 1., x_2 = 0.]], [2.28571428571429, [q1 = 2.28571428571429, q2 = 0., y_1 = 0.571428571428571, y_2 = 0.428571428571429]]$$

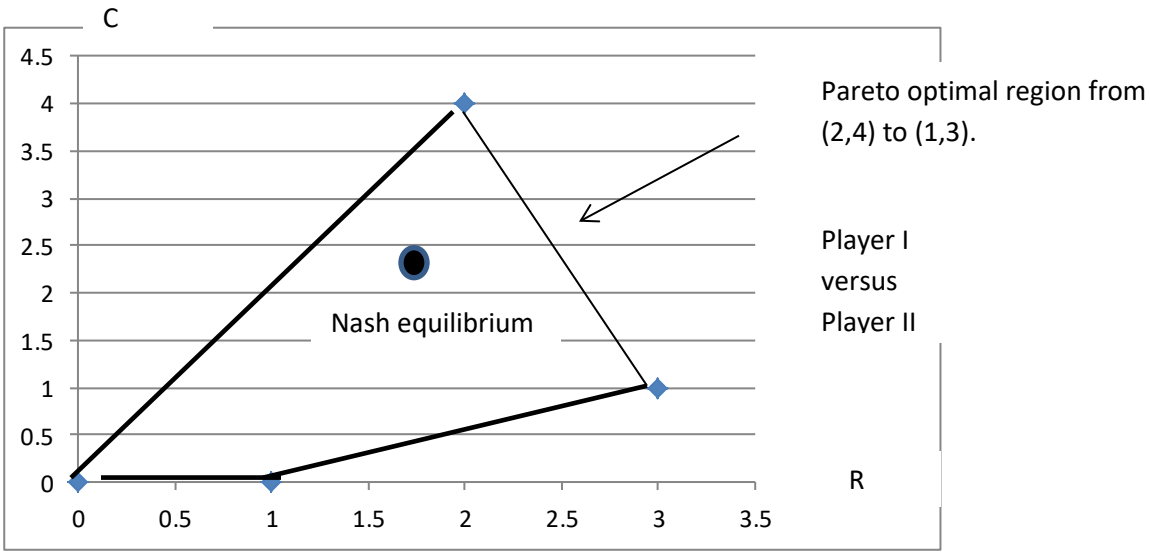

Figure 3. Payoff polygon and Pareto optimal region.

Nash Arbitration

According to Nash's bargaining problem *[1950], there is one and only one arbitration scheme that satisfies axioms 1 through 4. It is this: if the SQ =(x_0,y_0), then the arbitrated solution N is the point (x,y) in the polygon with x_>x_0 and y ≥ y_0 which maximizes the product (x-x_0)(y-y_0).*

These axioms according to Nash are:

Axiom 1: Rationality—the solution should be in the negotiation set.
Axiom 2: Linear Invariance. If either Rose or Colin's utilities are transformed by a positive linear function, the solution point should be transformed by the same function.
Axiom 3. Symmetry. If the polygon happens to be symmetric about the line of slope +1 through the SQ, then the solution point should be on this same line.
Axiom 4. Independence of Irrelevant Alternatives.

For more information, see a good discussion in Straffin [8] and Nash [7].

Further we assume in our examples that the SQ point is the security levels since every partial conflict game has security levels. Continuing our example where we previously found our security levels as 1 and 2.2857 we compute our solution.

```
NashArbitration := proc( slr, slc, x1, y1, x2, y2)
  with(LinearAlgebra) : with(Optimization) :
  objective := (x − slr)·(y − slc);
  in1 := (y − y1) − (y1 − y2)/(x1 − x2)·(x − x1) ;

  NLPSolve(objective, {in1 = 0, x ≥ slr, y ≥ slc, x ≤ x2, y ≤ y1},
  maximize);

end proc;
```

```
NashArbitration(1, 16/7, 2, 4, 3, 1);
```

```
[1.71428571428571441, [x = 2., y = 4.]]
```

The solution is the point (2,4) and value of the function to optimize is 1.7142857. We note that this point is an end point of the Pareto optimal line segment that lies in negotiations set. We provide another example where the solution is an interior point in the negotiations set.

Example 4. Given the following game from Straffin [8, p.102],

		Colin	
		C1	*C2*
Rose	*R1*	*(2,6)*	*(10,5)*
	R2	*(4,8)*	*(0,0)*

First, we find the security levels:

```
> PrudentialStrategies(ar, ac, A, B);
[3.33333333333333, [p1 = 3.33333333333333, p2 = 0., x1
    = 0.333333333333333, x2 = 0.666666666666667]], [6., [q1 = 6.,
    q2 = 0., y1 = 1., y2 = 0.]]
```

These are 10/3 for Rose and 6 for Colin. We then use these security levels in the Nash arbitration scheme.

```
> NashArbitration(10/3, 6, 4, 8, 10, 5);
[2.72222222222222010, [x = 5.66666666666667, y
    = 7.16666666666667]]
```

The Nash arbitration point is (5.666667, 7.1666667). The value of the function to be maximized is 2.72222.

Conclusions

We have shown how to use optimization to solve the Total Conflict and Partial Conflict games. We point out that we built many Maple procedures to assist with finding these results for both types of games. These templates are helpful to professors and students alike. With

these Maple procedures the number of strategies for each player is not limited by the solution methodology. The author will provide the MAPLE templates and procedures, upon request.

References

1. Barron, E.N. (2013) Game theory: An introduction. Hoboken, NJ: John Wiley & Sons.

2. Fox, W. (2008). Mathematical modeling of conflict and decision making: The writers' guild strike 2007–2008, Computers in Education Journal, 18(3), 2–11.

3. Fox, W. (2010). Teaching the applications of optimization in game theory's zero-sum and non-zero sum games, International Journal of Data Analysis Techniques and Strategies, 2(3), 258-284.

4. Fox, W. (2012). Mathematical Modeling with Maple. Boston, MA.: Cengage Publishers.

5. Gillman, R., & Housman, D. (2009). Models of Conflict and Cooperation, Providence, RI: American Mathematical Society.

6. Giordano, F., W. Fox, & S. Horton (2013) A First Course in Mathematical Modeling, 5th Ed. Boston, MA: Brooks-Cole.

7. Nash, J. (1950). The Bargaining Problem, Econometrica, 18, 155–162.

8. Straffin, P. (2004). Game Theory and Strategy. Washington, DC: Mathematical Association of America.

9. Winston, W. (2003). Introduction to Mathematical Programming, 4th Ed. Belmont, CA: Duxbury Press.

Biographical Information

William P. Fox is a professor at The Naval Postgraduate School in Monterey, California. He obtained his Ph.D. degree in Industrial Engineering and Operations Research from Clemson University and his M.S. degree in Operations Research from the Naval Postgraduate School. His research interests include modeling, optimization, game theory, and simulation. He has many conference presentations including: INFORMS, Mathematical Association of America Joint Annual Conference, Military Application Society (MAS), and the International Conference of Technology in Collegiate Mathematics (ICTCM). He has coauthored several books and over one hundred articles. He has previously taught at both West Point and Francis Marion University. He was the Director of both the High School Mathematical Contest in Modeling (HiMCM) and the collegiate Mathematical Contest in Modeling (MCM) and is currently the Past-President of the Military Applications Society of INFORMS.