A Web-Based Quadratic Bimatrix Game Optimization Model and Algorithms of Polynomial Complexity

Jonas Mockus,
Institute of Mathematics and Informatics,
Kaunas Technological University,
Vilnius Gedimino Technical University,
Akademijos 4, Vilnius LT-2600, Lithuania.
e-mail: jmockus@gmail.com

February 1, 2008

Abstract Web-based game optimization models are convenient examples for scientific cooperation and graduate studies of both the theory of games and the theory of optimization. This can be achieved by on-line investigation and modification of the game optimization models.

The aim of this paper is a polynomial time algorithm for a subset of quadratic bimatrix games where the Nash equilibrium exists in strictly mixed strategies. Strictly mixed means no pure strategies. This is a natural extension of the well known polynomial time direct search algorithm if the Nash equilibrium exists in pure strategies.

For search of Nash equilibrium in partly mixed strategies the heuristic algorithm of Quadratic Strategy Elimination is described. This algorithm is of polynomial time, too, and includes automatic testing thus just correct solutions are accepted.

The software as Java applet is integrated into the web-based system for scientific cooperation and graduate studies of optimization and theory of games. This helps readers to use their creative abilities by improving the presented software and by adapting it to their own needs and tastes.

The web-site *http* : *//pilis.if.ktu.lt/jmockus* includes this and accompanying optimization models.

Key Words: Game theory, Complexity theory, Heuristics, Education, Economics

Introduction

Traditional economical models describe the competition in markets. The models define such prices and other production parameters that satisfy the Nash equilibrium [16]. However, the market competition represents just a part of competitive economical and social activities. Another part is the inspection. Objectives of inspection are tax collection, property, health, and environment protection, e.t.c.
This paper investigates an example of quadratic bimatrix game model that describes the competition between inspectors and violators. We call these as Inspector Games (IG) The results illustrate how the theories of games [18],[1],[9],[10],[2] and global optimization[4], [11] can be applied to inspection problem defined as quadratic bimatrix game model and show the limitations imposed by high complexity of the problem and ways to avoid them.
The aim is to develop polynomial time algorithms for search of equilibrium in strictly mixed strategies of quadratic bimatrix games. That extends, in a sense, well known simple algorithms for search of equilibrium in pure strategies.

The game model of this paper is included as an example in the system for graduate studies and scientific collaboration in the internet environment in engineering optimization and market games [13], [5], [8], [7], [3], [12]. The system includes theory and software of different optimization models in 85,000 files:
$http://pilis.if.ktu.lt/\tilde{}jmockus$.
Published examples of this system is optimization of stock exchange model [5] and Walras competition model [6].

The possibility to run software developed by colleagues directly by Internet is essential for scientific collaboration and distance studies. We can test directly results of other researchers by running their software with our data. Therefore algorithms, software and results published in the scientific papers can be investigate independently. This possibility is not widely used yet. But the potential appears great.

In this sense Java applets are similar to proofs of theorems in mathematics since we test theorems by reading the proofs. The snapshots of graphical user interfaces are useful for testing the results, too. The snapshots help to do calculations exactly as authors intended.

In the Internet environment a platform independent language running

software on remote computers is needed. For example, Java, Perl, Python, PHP. Java is more efficient for large scale optimization problems.

1 Bimatrix Game (BG)

1.1 Complexity

The well known results of algorithm complexity show the limitations of exact solutions. That explains popularity of heuristic algorithms.
An important open problem in complexity theory is the existence of polynomial-time algorithms for computing equilibrium for subsets of bimatrix games. For games in extensive form, the reduced normal form may be exponentially large [21].
The Lemke-Howson algorithm [1] is the classical algorithm for the problem of finding a Nash equilibrium of a bimatrix game. It solves a linear complementary problem and provides a constructive and elementary proof of existence of an equilibrium.
The paper [19] presents a class of bimatrix games for which the Lemke-Howson algorithm takes, even in the best case, exponential time in the dimension of the game.

The book [14] shows that the computational effort required to solve a linear complementarity problem, by either of the two well known complementary pivot methods is not bounded above by a polynomial in the size of the problem. In was shown that to solve the problem of order n, either of the two methods goes through 2^n pivot steps before termination.
In this paper an algorithm is investigated that follows directly from the conditions of the Nash equilibrium. The algorithm is implemented using standard software of Linear Programming (LP). This algorithm defines equilibrium of quadratic bimatrix games in polynomial time if the equilibrium in strictly mixed strategies exists and interior point algorithm of LP is applied.

1.2 Utility Functions

The payoff function of the first player is expressed by a matrix $u(i, j)$ where $i = 1, ..., m$ denote moves (pure strategies) of the first player and $j = 1, ..., n$ represent moves of the second. The payoff of the second player is $v(i, j)$. The utility functions U and V of the first and second players are defined as

expected values of the payoffs $u(i,j)$ and $v(i,j)$.

$$U(x,y) = \sum_{i,j} x_i u(i,j) y_j, \tag{1}$$

and

$$V(x,y) = \sum_{i,j} x_i v(i,j) y_j. \tag{2}$$

Here $x = (x_i,\ i = 1,...,m)$ and $y = (y_j,\ j = 1,...,n)$ are probabilities (mixed strategies) of moves i and j.

In the Matrix Game $u(i,j) = -v(ij)$. In the Bimatrix Game $u(i,j) \neq -v(i,j)$. In the Quadratic Bimatrix Game $m = n$. Denote this game by a pair (U,V) in short.

Often the payoff functions are defined by some parameters. For example, Inspector Game models are defined by three main parameters: the probability p_i to detect a violation if it happens in the area i, the probability q_i of the violation, and the payoff g_i of the violation.
Two "anti-corruption" parameters can be regarded, too. The penalty ω for unauthorized deal and a probability p_ω of the penalty ω.

2 Search for Equilibrium

2.1 Necessary and Sufficient Conditions

Definition 1. *A game strategy x: $x_i \geq 0,\ \sum_{i=1}^m x_i = 1$ is called the mixed strategy.*
A game strategy x: $x_i > 0,\ \sum_{i=1}^m x_i = 1$ is called the strictly mixed strategy.
A game strategy x: $x_i = \{0,1\},\ \sum_{i=1}^m x_i = 1$ is called the pure strategy.

Theorem 1. *For a pair (x^*, y^*) to be a Nash equilibrium in strictly mixed strategies of the quadratic bimatrix game it is necessary and sufficient that there exist real numbers (α^*, β^*) such that $(\alpha^*, \beta^*, x^*, y^*)$ satisfies the system*

$$Uy = \alpha I, \tag{3}$$

$$xV = \beta I, \tag{4}$$

$$x > 0, y > 0, Ix = 1, Iy = 1, \tag{5}$$

where I denotes the unit vector.

Proof. From the well known [1] necessary and sufficient conditions of Nash equilibrium for general bimatrix games (U, V) follows that

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_i u_{i,j} y_j = \alpha, \tag{6}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_i v_{i,j} y_j = \beta \tag{7}$$

$$\sum_{j=1}^{n} u_{i,j} y_j \leq \alpha,\ i = 1, ..., m, \tag{8}$$

$$\sum_{i=1}^{m} x_i vi, j \leq \beta,\ j = 1, ..., n \tag{9}$$

$$x_i > 0, y_j > 0, \sum_{i=1}^{m} x_i = 1, \sum_{j=1}^{n} y_j = 1.$$

From (3), (4)

$$\sum_{j=1}^{m} u_{i,j} y_j = \alpha,\ i = 1, ..., m, \tag{10}$$

$$\sum_{i=1}^{m} x_i v_{i,j} = \beta,\ j = 1, ..., m \tag{11}$$

From (10), (11), (6), (7)

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_i u_{i,j} y_j = \sum_{i=1}^{m} x_i \sum_{j=1}^{n} u_{i,j} y_j = \sum_{i=1}^{n} x_i\ \alpha = \alpha \sum_{i=1}^{n} x_i = \alpha \tag{12}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_i v_{i,j} y_j = \sum_{j=1}^{m} y_j \sum_{j=1}^{n} x_i v_{i,j} = \sum_{j=1}^{n} y_j\ \beta = \beta \sum_{j=1}^{n} y_j = \beta. \tag{13}$$

That proves the sufficiency of (3), (4), (5).

To prove the necessity of (3), (4), (5) assume that

$$\sum_{j=1}^{m} u_{1,j} y_j = \alpha - \epsilon,\ \epsilon > 0, \tag{14}$$

$$\sum_{j=1}^{m} u_{i,j} y_j = \alpha,\ i = 2, ..., m, \tag{15}$$

$$\tag{16}$$

Then

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_i u_{i,j} y_j = (\alpha - \epsilon)x_1 + \alpha \sum_{i=2}^{m} x_i = \alpha - \epsilon x_1 < \alpha. \tag{17}$$

That proves the necessity.

2.2 Direct Testing

The equilibrium can be tested directly by the conditions of Nash equilibrium. First define the "contract" wins U^*, V^* assuming that both players keep the contract solution (x^*, y^*)

$$U^* = \sum_{i,j} x_i^* u(i,j) y_j^*, \tag{18}$$

$$V^* = \sum_{i,j} x_i^* v(i,j) y_j^*. \tag{19}$$

Then define the maximal wins U_{max}, V_{max} of a player assuming that other player keeps the contract solution (x^*, y^*)

$$U_{max} = \max_x \sum_{i,j} x_i u(i,j) y_j^*, \tag{20}$$

$$V_{max} = \max_y \sum_{i,j} x_i^* v(i,j) y_j. \tag{21}$$

The contract wins U^*, V^* are compared with the values U_{max}, V_{max}, and if

$$U^* \geq U_{max},\ V^* \geq V_{max}. \tag{22}$$

the contract solution $x_i^* > 0,\ y_j^* > 0$ is recorded as equilibrium strategy.

2.3 Irrelevant Fraud Algorithm (IF)

Expressions (10), (11), (5) is a system of $2m + 2$ linear equations that can be solved by standard algorithms of linear algebra. However the Linear Programming (LP) is convenient for analysis of results.

In LP all the variables should be non-negative. Thus we express all the original variables as differences of two non-negative LP variables

$$U = u_1 - u_2,\ V = v_1 - v_2,\ u_1 \geq 0,\ u_2 \geq 0, v_1 \geq 0, v_2 \geq 0,$$
$$y_j = y_j^1 - y_j^2,\ y_j^1 \geq 0,\ y_j^2 \geq 0,\ j = 1, ..., m,$$
$$x_i = x_i^1 - x_i^2,\ x_i^1 \geq 0,\ x_i^2 \geq 0,\ i = 1, ..., m. \tag{23}$$

Then from expressions (10), (11), (5) follows this LP problem

$$\max_{x,y,u,v}(u_1 - u_2 + v_1 - v_2), \tag{24}$$

$$\sum_{j=1}^{m} u(i,j)y_j = u_1 - u_2,\ i = 1, ..., m, \tag{25}$$

$$\sum_{i=1}^{m} v(i,j)x_i = v_1 - v_2,\ j = 1, ..., m, \tag{26}$$

$$\sum_{j=1}^{m} y_j = 1, \tag{27}$$

$$\sum_{i=1}^{m} x_i = 1, \tag{28}$$

$$\tag{29}$$

where $x = (x_1, ..., x_m),\ y = (y_1, ..., y_m)$,
$u = (u_1, u_2),\ v = (v_1, v_2)$. Here variables x_i and y_j can be negative. That provides some additional information about the problem when no positive solution exists.

Theorem 2. *If exists, the Nash equilibrium in strictly mixed strategies of quadratic bimatrix game can be defined by an algorithm of polynomial complexity.*

Proof. The Interior Point algorithms of LP problems are of polynomial complexity [15]. That proves the theorem assuming that the Interior Point implementation of IF algorithm is used.

That is the theoretical result. In large scale practical problems the simplex algorithm of LP problems can be more efficient since its average complexity is low. The simplex algorithms of LP are of exponential complexity in the worst case and of low degree polynomial complexity in average [20].

Thus, if exists, the positive IF solution

$$x_i > 0,\ y_j > 0,\ i, j = 1, ..., m \tag{30}$$

defines the Nash equilibrium for the quadratic bimatrix game (1)(2). The problem is that positive solutions of IF algorithm not always exists. An example is the Nash equilibrium in pure strategies (32). The Direct Search algorithm (DS) can be applied for a search of equilibrium in pure strategies. If that fails, too, then the heuristic Quadratic Strategy Elimination (QSE) is used.

2.4 Direct Search Algorithm (DS)

It is well known [1] that, in randomly generated large bimatrix games, the proportion of games having k pure strategy equilibrium is defined by the Poisson distribution

$$e^{-1}/(k!) \tag{31}$$

That means that roughly two-thirds of randomly generated bimatrix games have an equilibrium point in pure strategies. This result is proved for randomly generated bimatrix games. Example 3.2.1 shows that happens in small quadratic bimatrix games, too.
Denote by $I(j)$ a set of maximal elements of column j of matrix $u(i,j)$. Denote by $J(i)$ a set of maximal elements of row i of matrix $v(i,j)$. Denote unions of these sets $I = \cup_j I(j)$ and $J = \cup_i J(i)$. Intersection of these unions

$$IJ = I \cap J, \tag{32}$$

defines equilibrium in pure strategies. If IJ is empty then we need mixed strategies [18].

2.5 Quadratic Strategy Elimination Algorithm (QSE)

The quadratic strategy elimination algorithm includes both IF and DS algorithms. In addition, it eliminates irrelevant rows and columns:

1. obtain a solution (x^*, y^*) of LP problem (24)- (23),
   if a positive solution of the system is found, go to step 7

2. search for equilibrium in pure strategies by the Direct Search algorithm (32)
   if a solution is found, go to step 7

3. reduce the system of LP problem by eliminating both the column $j = k$ and the raw $i = k$
   if $y_k \leq 0$ or/and $x_k \leq 0$,

4. set to zero both the variables $y_k = 0$ and $x_k = 0$

5. if a positive solution of the reduced system is found, go to step 8

6. if the reduced system is not empty, go to step 3

7. test conditions (22),
   if no - print 'No solution found' ,
   if yes - go to Finish 8,

8. record the solution $x_i^*,\ y_j^*$ as an equilibrium strategy.

2.6 Preventing Corruption

The game will not be played by rules if the players may win more by breaking them. Unauthorized deals (for example, bribes) are common tools for breaking the rules of non-zero-sum games where $v(i,j) \neq -u(i,j)$. An example of such deal is sharing the violator win between the inspector and violator. Values of the optimal deal $(\bar{u}, \bar{v})$ are defined by the Nash bargaining conditions [18]

The optimal deal $(\bar{u}, \bar{v})$ depends on the set of feasible deals D,
and on guaranteed wins defining what the players will get if the deal fails

$$u^* = max_x\ min_y U(x,y),$$
$$v^* = max_x\ min_y V(x,y). \tag{33}$$

Here
u^* is the maximal expected guarantee win of the first player,
v^* is that of the second player.
In the Nash deal the first player gets $\bar{u}$ and the second obtains $\bar{v}$.

If exists a pair $(u,v) \in D$, such that $u > u^*,\ v > v^*$, then the optimal deal

$$(\bar{u}, \bar{v}) = arg\ max_{u,v}(u - u^*)(v - v^*), \tag{34}$$

where

$$(u, v) \in D,\ u \geq u^*,\ v \geq v^* \tag{35}$$

If the sum of expected wins of both players is restricted by c then

$$D = \{(u, v) : u + v \leq c\}, \tag{36}$$

and the Nash deal

$$\begin{gathered}(\bar{u}, \bar{v}) = \\ ((c + u^* - v^*)/2,\ (c + v^* - u^*)/2)\end{gathered} \tag{37}$$

An obvious way to prevent unauthorized deals is by introducing an additional parameter w defining the expected penalty that makes the deal unprofitable. The deal profits of players

$$\begin{aligned} u_d &= \bar{u} - u^* = 1/2(c + u^* - v^*) - u^*, \\ v_d &= \bar{v} - v^* = 1/2(c + v^* - u^*) - v^* \end{aligned} \tag{38}$$

We see that the profits are equal. Thus the minimal expected penalty

$$w^* = 1/2\ (c - u^* - v^*) \tag{39}$$

Deal fails if the expected penalty $w > w^*$. Here $w = \omega\ p_\omega$ where ω is the deal penalty and p_ω is the probability of deal detection.
Assuming, that the deal of some IG is arranged before the game

$$c = \max_i\ g_i q_i \tag{40}$$

Here c is the maximal expected payoff to be divided between the bargaining players.

2.7 Comparing Algorithms

The well known general exact algorithms are of exponential complexity [14] and [19].
The advantage of algorithms of this paper comparing with the algorithms [1], [17] is the polynomial complexity if equilibrium in strictly mixed strategies exists and the Interior Point algorithm of Linear Programming is used.

Therefore the important problem of future investigation is evaluation of the proportion of quadratic bimatrix games with equilibrium in strictly mixed strategies. Theoretical definition of this proportion is difficult. Thus some large scale experimentation is needed. Additional task of experimental investigation is to define problems solvable by the heuristic part of QSE algorithm. That is just short remark. The large scale computer experiments of randomly generated quadratic bimatrix problems is the subject of separate investigation.

3 Examples of Inspector Games

3.1 Poaching Game

Denote by $x = (x_1, ..., x_m),\ x_i \geq 0,\ \sum_i x_i = 1$ the inspection vector and by $y = (y_1, .., y_m),\ y_j \geq 0,\ \sum_j y_j = 1$ the violation vector. Here x_i denotes the inspection probability of the area i and y_j means the violation probability in the area j. Denote by $u(i, j)$ the inspector's payoff function when the object i is inspected and the object j is violated. Denote by $v(i, j)$ the poacher's payoff function when the object i is inspected and the object j is violated. Functions $U(x, y)$ and $V(x, y)$ denote the inspection and violation expected utility functions using inspection and violation vectors x, y. These vectors define probabilities of inspection and violation.

For example,

$$u(i,j) \quad = \quad \begin{cases} p_i g_i q_i, & \text{if } i = j \\ 0, & \text{otherwise,} \end{cases} \tag{41}$$

and

$$v(i,j) \quad = \quad \begin{cases} -q_j p_i g_j + (1 - p_i) q_j g_j, & \text{if } i = j \\ q_j g_j, & \text{otherwise.} \end{cases} \tag{42}$$

Here p_i is the probability of detecting the violation if it happens in the area i. A number q_i is the probability of the violation in the area i and g_i is the payoff (potential) of the violation in the area i.

Expression (41) means that if the violation is completed and detected (the prey is killed and a poacher is caught) then the inspector's premium is equal to the payoff of violation (the value of the killed prey)is simple. Expression (42) shows that if the violation is completed and detected, the

violator's payoff is negative. The payoff is positive, if the violation is not detected.

The utility functions at given inspection and violation vectors x and y

$$U(x,y) = \sum_{i,j} x_i u(i,j) y_j, \tag{43}$$

$$V(x,y) = \sum_{i,j} x_i v(i,j) y_j. \tag{44}$$

Here $u(x,y) \neq v(x,y)$, thus the inspection model is a bimatrix game [1].

3.1.1 Explicit Solution

There exists explicit IF solution for a subset of IG. These solutions are useful for software testing.
Suppose, for example, that $q_i = g_i = 1$. Then from (41)(42)

$$u(i,j) \;\; = \;\; \begin{cases} p_j, & \text{if } i = j \\ 0, & \text{otherwise,} \end{cases} \tag{45}$$

and

$$v(i,j) \;\; = \;\; \begin{cases} -p_i + (1-p_i), & \text{if } i = j \\ 1, & \text{otherwise.} \end{cases} \tag{46}$$

From here and expressions(10), (11), (5)

$$p_j y_j = U, \;\; j = 1, ..., m \tag{47}$$

$$\sum_{i \neq j} x_i + (1 - 2p_j) x_j = V, \;\; j = 1, ..., m \tag{48}$$

$$\sum_j y_j = 1, \;\; \sum_i x_i = 1, \;\; y_j \geq 0, \;\; x_i \geq 0$$

The solution is simple

$$y_i = x_i = 1/(p_i \sum_{k=1}^{n} 1/p_k), \;\; i = 1, ..., n. \tag{49}$$

Note, that

$$x_i > 0, \;\; y_j > 0, \;\; i, j = 1, ..., m. \tag{50}$$

We see that if $g_i = q_i = 1$ then the IG problem (41)(42) is solved by the Irrelevant Fraud (IF) algorithm. The same is true if $g_i = g,\ q_i = q,\ i = 1, ..., n$.

If no positive solution exist then we search for the equilibrium in pure strategies by the Direct Search) algorithm (32). That complements the IF algorithm (27) - (23) and defines the equilibrium in pure strategies, if it exists. The next step is to apply the QSE algorithm.

3.1.2 Corrupt Inspector

Suppose that $g_i = q_i = 1$ and the sum of expected wins is restricted by $c = \max_i\ g_i q_i = 1$.
From (49)

$$x_1^* = y_1^* = p_2/(p_1 + p_2),$$
$$x_2^* = y_2^* = p_1/(p_1 + p_2).$$

Then from (1), (2), (45), and (46)

$$u^* = p_1 p_2/(p_1 + p_2),$$
$$v^* = 1 - 2p_1 p_2/(p_1 + p_2) = 1 - 2u^*.$$

$$w^* = 1/2(1 - u^* - v^*) = 1/2(1 - u^* - 1 + 2u*) = 0.5u^* \tag{51}$$

If, for example, $p_1 = p_2 = 0.5$ then the expected penalty $w^* = 0.125$.
Figure 1 shows input data for $m = n = 5$,

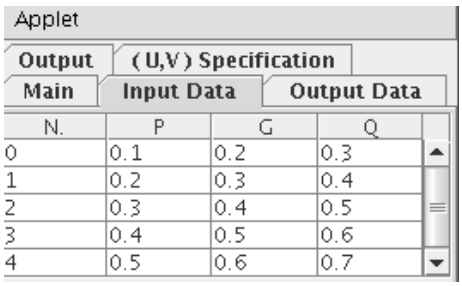

Figure 1: Parameter table of IG for $m = n = 5$

Figure 2 shows optimal strategies of QSE for a poacher game. In this software version ranger's mixed strategies are denoted by $Y0$ and poacher's - by

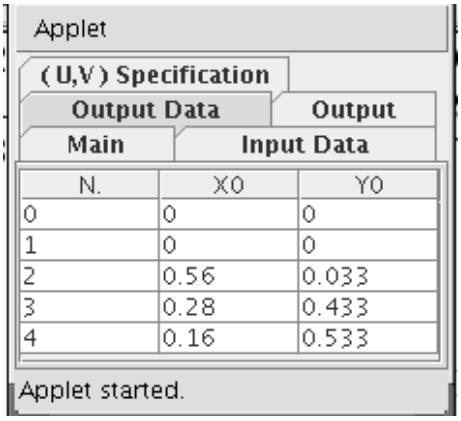

Figure 2: Optimal strategies of QSE for poacher game.

$X0$. Note that the corresponding variables in equations 1 and 2 are denoted differently: ranger's - mixed strategy by x and poacher's - by y.
Figure 3 shows detail output of calculations including the expected payoffs and the optimal deal. All three figures are by the experimental software described in section 4.

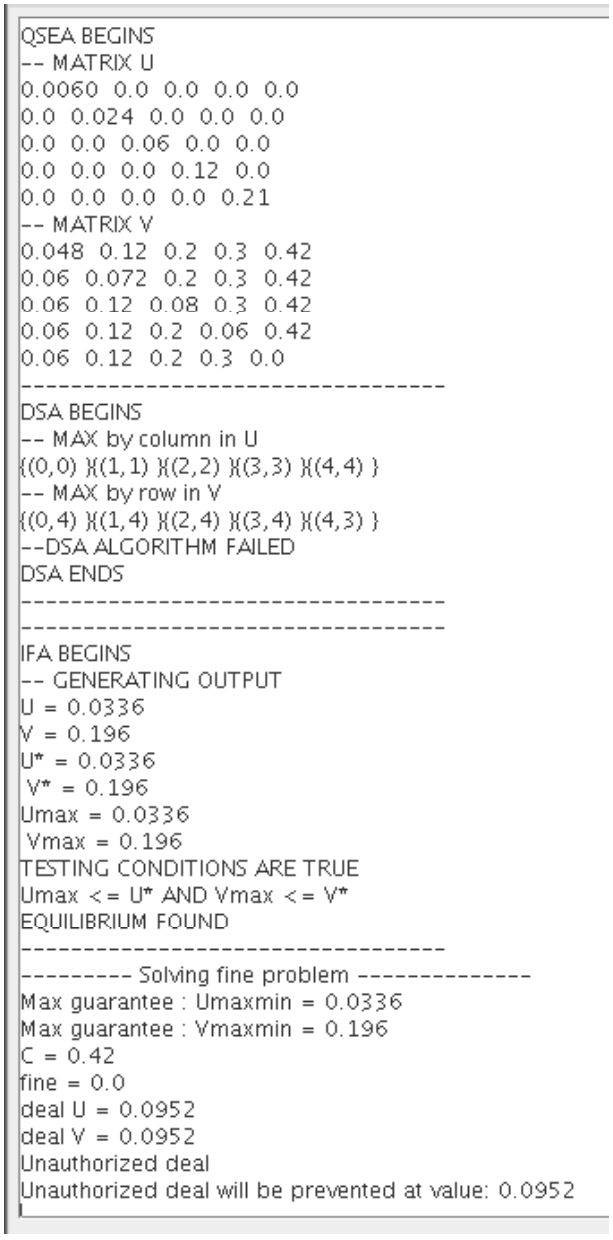

Figure 3: Output of QSE for poacher game including payoff matrix and deal.

3.2 Security Game

A security system protects some object. Denote by $x = (x_1, ..., x_m)$, $x_i \geq 0$, $\sum_i x_i = 1$ the protection vector and by $y = (y_1, .., y_m)$, $y_j \geq 0$, $\sum_j y_j = 1$ the violation vector. Here x_i denotes the part of security funds designated to protect the area i and y_j means the violation probability of the area j. Denote by $u(i, j)$ the security payoff function when the area i is protected and the area j is violated. Denote by $v(i, j)$ the violation payoff function when the area i is protected and the area j is violated. Utility functions $U(x, y)$ and $V(x, y)$ denote expected values of security and violation payoff functions using protection and violation vectors x, y.

These vectors define the distribution of protection funds and violation probabilities. An example of simplified payoff functions: the security payoff function

$$u(i,j) = \begin{cases} p_i g_i - (1-p_i) g_i q_i, & \text{if } j = i \\ 0, & \text{if } j \neq i, \end{cases} \tag{52}$$

Here $p_i g_i$ is the expected reward of security team that protects the area i for preventing the violation and arresting the violators. Denote by p_i the probability of this event. A number g_i is the value of protected goods.
Denote by q_i the probability of completing the violation (for example, stealing goods) in the area i. The expected penalty for bad protection is defined by the product $(1-p_i)g_i q_i$. The payoff function of security team protecting area i is zero, is no violation happens in this area $i \neq j$.
Then the violation payoff function

$$v(i,j) = \begin{cases} -p_i b + (1-p_i) q_i g_i, & \text{if } j = i \\ q_j g_j, & \text{if } j \neq i. \end{cases} \tag{53}$$

Here $p_i b$ is the penalty of attempted and detected violation where b is a large number. The expected profit of successful violation of the protected area i is defined by the product $(1-p_i)q_j g_j$ and the expected profit of successful violation of the unprotected area j is $q_j g_j$.

Assume for simplicity that probability of protection p_i is equal to the part of security funds x_i. This is the linearization of the non-linear security function $p_i = p_i(x_i)$.

Expression (52) means that if the violation is completed and detected (goods are not stolen and the thief is caught) in the area i then the reward of security is equal to the value of the protected goods g_i. Expression (53)

shows that if the violation is detected, the violator penalty is a large number b. The violators payoff is equal to the value g_i of stolen goods if the crime in the area i is successfully completed and not detected.

The utility functions at given inspection and violation vectors x and y

$$U(x,y) = \sum_{i,j} x_i u(i,j) y_j, \tag{54}$$

and

$$V(x,y) = \sum_{i,j} x_i v(i,j) y_j. \tag{55}$$

Here $u(x,y) \neq v(x,y)$.

3.2.1 Numerical Solution

The following simple numerical example gives some "feeling" of the security game. Suppose that $g_1 = g_2 = g_3 = 1$,
$q_1 = q_2 = 0.1,\ q_3 = 0.01$ and $p_1 = 0.1,\ p_2 = 0.2,\ p_3 = 0.3$.
Then from (41)(42)

$$u(i,j) = \begin{vmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.29 \end{vmatrix}. \quad v(i,j) = \begin{vmatrix} 0.09 & 0.1 & 0.01 \\ 0.1 & 0.08 & 0.01 \\ 0.1 & 0.1 & 0.07 \end{vmatrix}. \tag{56}$$

No IF solution exists since from (24)- (23)

$$x_1 = 6.414\ x_2 = 3.207\ x_3 = -8.621 \tag{57}$$

$$y_1 = 1.0\ y_2 = 0.0\ y_3 = 0.0 \tag{58}$$

However, two DS solutions exist: $x_1 = 1,\ y_2 = 1$ and $x_1 = 1,\ y_3 = 1$.

4 Experimental Software

The experimental version, the QSE-software in short, is on the web[1] and is used for research cooperation and graduate studies. Several versions of QSE-software are implemented as Java applets. In the version denoted as

[1]See the problems "Bimatrix Game 4" in the web-site section "Discrete Optimization".

"updated example of bimatrix game optimization using LP, Java, QSE and 'Altruistic'[2] algorithms" the applet starts by clicking the line 'remote start, original 2007-02-07 version'. That is not secure mode thus the input is made writing text on the screen.

Figure 4 shows the data file 'rnd5-1.text' and values of anti-corruption

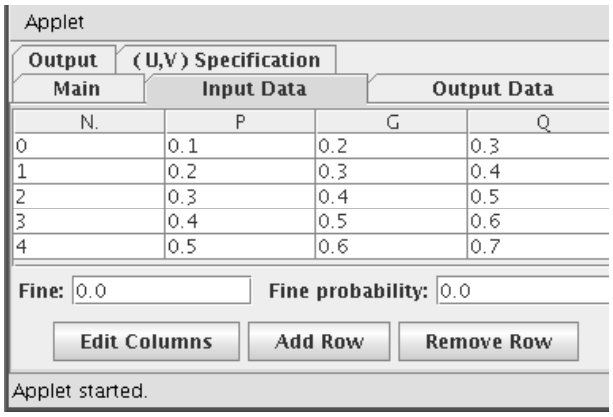

Figure 4: Data file.

parameters. To prevent unauthorized deals we enter 'Fine' and 'Fine's probability' defining the preferred penalties. The zero values mean no preferences thus the penalty is set by software.

Figure 5 shows how payoff function of the poacher game depend on parameters $Pi = p_i,\ GI = g_i, Qi = q_i$. Data can be recorded by downloading data file, for example 'data.txt', or by adding forest rows and entering their parameters P, Q, G.

The optimization starts by clicking the button *Calculate*.

A way of data input by files is to download the archive file 'remig05-19.tgz' and to start applet by the command line: 'appletviewer -J-Djava.security.policy=java.policy applet.html'[3].

Using the version denoted as "signed version of bimatrix game optimization" all the data is recorded by uploading an environmet file, for example:

```
V_EQUTION=-Qj*Pi*Gj+(1-Pi)*Qj*Gj;i\=j;\nQj*Gj;
U_EQUTION=Pi*Gi*Qi;i\=j;\n0;
DATA_SAVE_URL=file\:save_data.txt
```

[2]'Altruistic' means a heuristic algorithm, not recommended now.

[3]Here 'applet.html' defines a html script to start an IG problem.

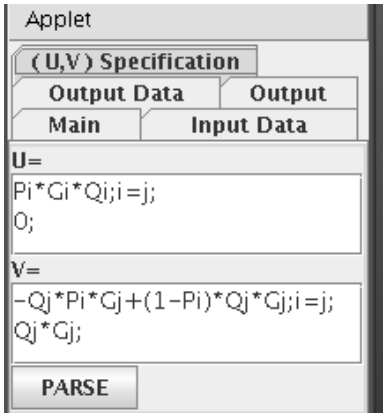

Figure 5: Payoff function of poacher game.

```
DATA_LOAD_URL=http\://pilis.if.ktu.lt/~jmockus/inspsign/src/data1.txt
VARIABLES=P G Q
RESULTS_SAVE_URL=file\:save_results.txt
```

Different browsers use different permission forms for signed applets. Figure 6 shows the Java security window of the "Opera" browser.
Input of data by files is needed for large scale experimentation.
Figure 7 shows an opening window defining the data file and the algorithm. The web-site is updated at the end of each semester.

4.1 Comparing with General Software Tools for Game Theory

The Gambit [10] is a popular software tool. That is a general software designed to solve different games. The advantage of Gambit is the implementation of exact algorithms and the convenient graphical user interface, based upon the 'wxWidgets' library.
In Figure 8 graphical user interfaces (GUI) of Gambit and QSA are compared. The exact Lemke algorithm [1] is implemented by Gambit and the QSE algorithm is realized by QSE-software. Upper tables show payoff matrices. Middle tables are results of exact algorithm and the lower tables show results of QSE. In this example of the poacher game the results are the same. In other examples the results may differ since QSE includes the heuristic part.

Figure 6: Applet security window.

The aim of the QSE-software is the web-based implementation of algorithms of polynomial complexity for a subset of quadratic bimatrix games. This is done by Java applets. A feature of Java as compared with other platform-independent languages is the 'just-in-time' compiler. That, and some new improvements in jdk-6 version, makes web-based optimization algorithms almost as fast as the stand-alone versions implemented in the efficient languages such as C++.
The QSE-software realizes a parser transforming payoff functions depending on several parameters into standard payoff matrices. In addition QSE-software implements algorithms for prevention of unauthorized deals.

5 Application for Distance Graduate Studies

The Optimal Inspection Model is simple. However, the model is based on fundamental results of games theory and operations research. That makes the model useful for studies of these topics. The software is designed as

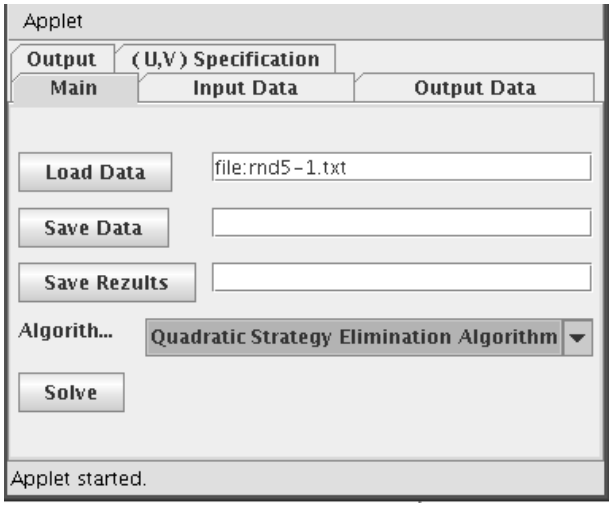

Figure 7: Opening window of IG.

an open-ended tool of research. Important task is to define a set of bimatrix games that can be solved by the polynomial algorithms. That includes defining proportion of randomly generated quadratic bimatrix games with Nash equilibrium in strictly mixed strategies. For equilibrium in pure strategies similar problem is solved by asymptotic formula (31). The problem for equilibrium in strictly mixed strategies seems to complicated for analytical solution. Thus extensive experimentation is needed. Using and developing the software students better understand applications of Nash equilibrium [16].

These properties are useful for graduate studies where research skills are important. Students can easily create new bimatrix game models using the "Parser" mode implemented in the software, see Figures 5. This way new features can be included and new situations investigated.

Implementation of the model as Java applet provides a cross-platform compatibility and makes the distances almost irrelevant.

The model is a part of a Web-based system of distance graduate studies: $http://pilis.if.ktu.lt/jmockus$. Now the system is used regularly for distance graduate studies in two Lithuanian universities: Kaunas Technological University and Vilnius Technical University. The system was used during international graduate studies including Lappeenranta University of Technology, SF-53851, Lappeenranta, Finland [3].

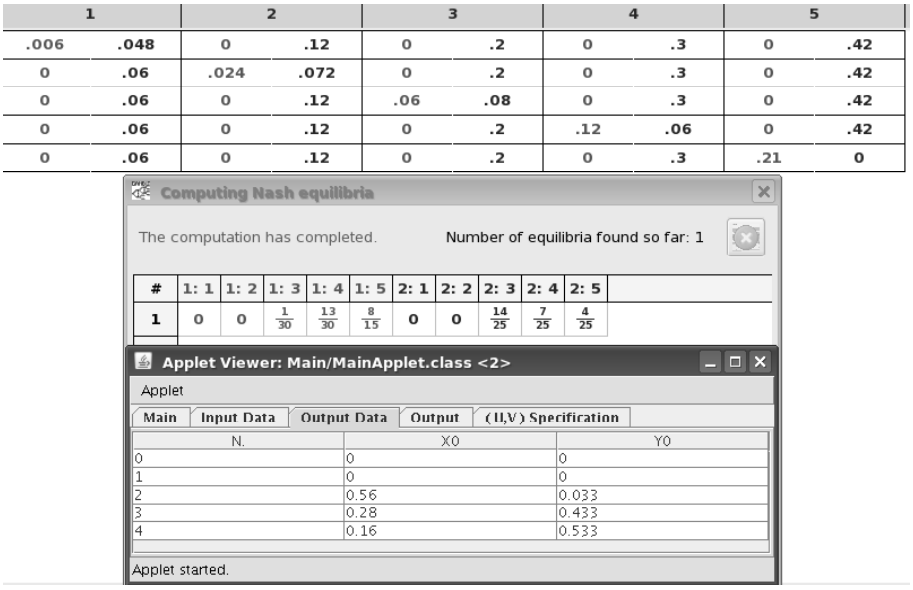

Figure 8: Comparison of Graphical User Interfaces of Gambit and QSE.

6 Concluding Remarks

The polynomial time algorithm is developed that defines Nash equilibrium in strictly mixed strategies for quadratic bimatrix games. The proportion of randomly generated quadratic bimatrix games with equilibrium in strictly mixed strategies remains open problem.

For quadratic bimatrix games in mixed strategies a heuristic algorithm is developed. The results are verified by direct testing.

The algorithms are included as Java applets in the web-based system for scientific cooperation and graduate studies. The source codes are included, too. High efficiency of new versions of Java enables large scale experimentation that is needed defining a subset of polynomially solvable bimatrix games.

References

[1] F. Forgo, Jeno Szep, and Ferenc Szidarovszky. *Introduction to the Theory of Games*. Kluwer Academic Publishers, 1999.

[2] Games. *Game theorists who have received the Nobel Prize*. http://lcm.csa.iisc.ernet.in/gametheory/nobel.html.

[3] Matti Heilio and Jonas Mockus. Web-based system for graduate studies - optimization, games and markets. The 14th European Conference on Mathematics for Industry (ECMI 2006), the UNIVERSITY CARLOS III DE MADRID, July 10-14, Madrid, Spain, 2006.

[4] R. Horst and P. Pardalos. *Handbook of Global Optimization*. Kluwer Academic Publishers, Dordrecht/Boston/London, 1995.

[5] J.Mockus. Stock exchange game model as an example for graduate level distance studies. *Computer Applications in Engineering Education*, 10:229–237, 2003. ISSN 1061-3773.

[6] J.Mockus. Walras competition model, an example of global optimization. *Informatica*, 15:525–550, 2004.

[7] J.Mockus. Investigation of examples of e-education environment for scientific collaboration and distance graduate studies, part 1. *Informatica*, 17:259–278, 2006.

[8] J.Mockus. A system for distance studies and applications of metaheuristics. *Journal of Global Optimization*, 35:637–651, 2006.

[9] Richard D. McKelvey and Andrew McLennan. Computation of equlibria in finite games. In *Handbook of Computational Economics*. 1996.

[10] Richard D. McKelvey, Andrew M. McLennan, and Theodore L. Turocy. Gambit: Software tools for game theory, version 0.2005.06.13. http://econweb.tamu.edu/gambit/, 2005.

[11] J. Mockus. *A Set of Examples of Global and Discrete Optimization: Application of Bayesian Heuristic Approach*. Kluwer Academic Publishers, 2000. ISBN 0-7923-6359-0.

[12] Jonas Mockus. Development and investigation of a system for distance graduate studies and research cooperation in the internet evironment; applications in lithuanian universities. In *Modelling and simulation of business systems, May 13-14, BaltORS*, pages 191–2003, Vilnius, Lithuania, 2003.

[13] Jonas Mockus. A system for distance graduate studies and research cooperation in the internet evironment and applications in lithuanian universities. web-site:. $http://pilis.if.ktu.lt/\tilde{}jmockus$, 2007.

[14] K.G. Murty. *Linear Complementarity, Linear and Nonlinear Programming*. Heldermann Verlag, Berlin, 1988.

[15] K.G. Murty. A new practically efcient interior point method for lp. *Algorithmic Operations Research*, 1:3–19, 2006.

[16] J. Nash. Equilibrium points in n-person games. *Proc. Nat. Acad. Sci. USA*, 36:48–49, 1950.

[17] O.R.Lab. Operations research program library, vol.2, ver. 6, bigame. http://www.orlab.org/software/or_prog/or2/index.html.

[18] G. Owen. *Game Theory*. W.B. Saunders, Philadelphia, PA, 1968.

[19] R. Savani and B. von Stengel. Exponentially many steps for finding a nash equilibrium in a bimatrix game. In *Foundations of Computer Science, 2004, Proceedings. 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS'04)*, pages 258 – 267, Rome, Italy, October 17-19, 2004., 2004.

[20] S. Smale. On the average number of steps of the simplex method of linear programming. *Mathematical Programming*, 27:241–267, 1983.

[21] Bernhard von Stengel. Computing equilibria for two-person games. In *Handbook of Game Theory, Vol. 3*. North Holland, Amsterdam, 1998.