Computer Science Journal of Moldova, vol.**26**, no.1(76), 2018

Parallel algorithm to find Bayes-Nash solution to the bimatrix informational extended game

Boris Hancu, Anatolie Gladei

Abstract

We propose to use the new methodology for solving the complete and perfect information bimatrix game. To solve the games of these type we construct the incomplete and imperfect information game generated by the informational extended strategies. Then we construct associated Bayesian game with non-informational extended strategies. For a HPC cluster computing system with shared and distributed memory, we constuct a parallel algorithm for computing Bayes-Nash solutions to the bimatrix informational extended games. The complexity and time performance analysis of the algorithm are provided.

Keywords: game, strategy, Nash equilibrium, Bayes-Nash solution, parallel algorithm, time complexity, communication complexity.

1 Bayes-Nash solutions to the bimatrix informational extended games

1.1 Bimatrix informational extended games

We consider the bimatrix game in the following strategic form

$$\Gamma = \langle I, J, A, B \rangle , \tag{1.1.1}$$

where $I = \{1, 2, .., n\}$ is the line index set (the set of strategies of the player 1), $J = \{1, 2, .., m\}$ is the column index set (the set of strategies of the player 2) and $A = ||a_{ij}||_{i\in I}^{j\in J}$, $B = ||b_{ij}||_{i\in I}^{j\in J}$ are the payoff matrices of player 1 and player 2, respectively. All players know exactly the payoff matrices and the sets of strategies. Players maximize

their payoffs. Denote by $(1 \leftrightarrows 2)$ the following "informational decision making model" [1,2]: the player 1 knows exactly the value of the strategy chosen by the player 2, as well as, simultaneously, the player 2 knows exactly the value of the strategy chosen by the player 1. So we will analyze the game (1.1.1) in complete information (the players know exactly the normal form of the game) and $(1 \leftrightarrows 2)$-perfect information over the sets of pure strategies.

The conditions described above stipulate that we can use the set of informational extended strategies $\mathbf{i} : J \to I$, so that $\forall j \in J$, $\mathbf{i}(j) \in I$ of the player 1 and $\mathbf{j} : I \to J$, so that $\forall i \in I$, $\mathbf{j}(i) \in J$ of the player 2. According to [3] we can describe the informational extended strategies in bimatrix game as follows: $\mathbf{i} = i_1 i_2 ... i_j ... i_m$ and $\mathbf{j} = j_1 j_2 ... j_i ... j_n$, where the element $i_j = \mathbf{i}(j)$ (respectively $j_i = \mathbf{j}(i)$) of this number string means the following: if the player 2 (respectively the player 1) chooses the column $j \in J$ (chooses the line $i \in I$), then the player 1 (the player 2) will choose the line $i_j \in I$ (will choose the column $j_i \in J$). So the number string $\mathbf{i}$, respectively $\mathbf{j}$, is called the informational extended strategy of the player 1, respectively informational extended strategy of the player 2. The sets $\mathbf{I} = \{\mathbf{i} = i_1 i_2 ... i_j ... i_m\}$, where i go through the values from I and $\mathbf{J} = \{\mathbf{j} = j_1 j_2 ... j_i ... j_n\}$, where j go through the values from J, are called the sets of informational extended strategies of the player 1, respectively of the player 2. The $(\mathbf{i}, \mathbf{j})$ is called the informational extended strategy profile. The set $\widetilde{I} = \{i_j \in I : i_j \neq i_k, \forall j, k \in J, j \neq k\} \subseteq I$, respectively $\widetilde{J} = \{j_i \in J : j_i \neq j_r \forall i, r \in I, i \neq r\} \subseteq J$, is the set of informational non extended strategies of the player 1, respectively 2, generated by the informational extended strategies $\mathbf{i}$, respectively $\mathbf{j}$.

It should be mentioned that the players do not know the informational extended strategies of each other and from this point of view we can consider that the game has imperfect information structure over the sets of the informational extended strategies.

Denote by $Game\,(1 \leftrightarrows 2)$ the bimatrix game in the informational extended strategies, described above. Remark that the notation $Game\,(1 \leftrightarrows 2)$ does not represent the normal form. This game is in imperfect information on the set of informational extended strategies,

but because we do not know yet the normal form, we can not say if this game is in complete or incomplete information. For the game in informational extended strategies it is very difficult to construct utility matrices. So we can not answer the following question: for any informational extended strategy profile $(\mathbf{i},\mathbf{j})$ which **element of the matrix** A and B should be considered as a payoff value of the player 1 and 2 ? For example, consider the following bimatrix game $A=\begin{pmatrix}3 & 5 & 4\\ 6 & 7 & 2\end{pmatrix}$, $B=\begin{pmatrix}0 & 5 & 1\\ 4 & 3 & 2\end{pmatrix}$ in complete and $(1\leftrightarrows 2)$-perfect information and suppose that the informational extended strategy of the player 1 is $\mathbf{i}=1_1 1_2 2_3$, and for the player 2 is $\mathbf{j}=1_1 2_2$. For strategy profile $(\mathbf{i},\mathbf{j})\equiv(1_1 1_2 2_3, 1_1 2_2)$ we can not determine which **elements** in matrices A and B can be considered as payoff values for players 1 and 2.

Thus, in order to solve games in informational extended strategies, we propose to use the new methodology described in [3]. We will briefly present one of these methodologies.

1.2 The incomplete and imperfect information game generated by the informational extended strategies

As it was mentioned in [3] any fixed strategy profile $(\mathbf{i},\mathbf{j})$ in informational extended strategies generates a couple of matrices $A(\mathbf{i},\mathbf{j})=||\mathfrak{a}_{ij}||_{i\in I}^{j\in J}=||a_{i_j j_i}||_{i\in I}^{j\in J}$, $B(\mathbf{i},\mathbf{j})=||\mathfrak{b}_{ij}||_{i\in I}^{j\in J}=||b_{i_j j_i}||_{i\in I}^{j\in J}$ that represent the utility of the players in informational non extended strategies generated by the informational extended strategies $\mathbf{i}$, respectively $\mathbf{j}$. For strategy profile $(\mathbf{i},\mathbf{j})$ the elements $\mathfrak{a}_{ij}\equiv a_{i_j j_i}$, $\mathfrak{b}_{\mathbf{ij}}\equiv b_{i_j j_i}$ reprezent the payoff of the player 1 and of the player 2, respectively, if player 1 chooses the line $i\in I$ and the player 2 chooses the column $j\in J$ in the complete and $(1\leftrightarrows 2)$-perfect information bimatrix game. As the players do not know what informational extended strategies are chosen by the other partners, then the player 1, respectively player 2, will have a possible utility matrices from the following set of matrices $\{A(\mathbf{i},\mathbf{j})\}_{\mathbf{i}\in\mathbf{I}}^{\mathbf{j}\in\mathbf{J}}$, respectively $\{B(\mathbf{i},\mathbf{j})\}_{\mathbf{i}\in\mathbf{I}}^{\mathbf{j}\in\mathbf{J}}$. The game of this type is *the game in incomplete information* because neither player 1 nor player 2 knows exactly

which matrix from the mentioned set of matrices will be his utility.

So the game $Game\,(1 \leftrightarrows 2)$ in imperfect information on the set of informational extended strategies generates the following incomplete and imperfect information game on the set of informational non extended strategies: the strategies of the player 1 are $I = \{1, 2, ..., n\}$ and of the player 2 are $J = \{1, 2, ..., m\}$; the payoff matrix of the player 1 is one of the matrices from the set $\{A(\mathbf{i}, \mathbf{j})\}_{\mathbf{i}\in\mathbf{I}}^{\mathbf{j}\in\mathbf{J}}$ and the payoff matrix of the player 2 is one of the matrices from the set $\{B(\mathbf{i}, \mathbf{j})\}_{\mathbf{i}\in\mathbf{I}}^{\mathbf{j}\in\mathbf{J}}$.

Finally, using the informational extended strategies in the game (1.1.1) with complete and $(1 \leftrightarrows 2)$-perfect information we obtain the following normal form of the incomplete (asymmetric) and imperfect information game

$$\widetilde{\Gamma} = \left\langle \{1, 2\}, I, J, \left\{ AB(\mathbf{i}, \mathbf{j}) = \left\| \left(a_{i_j j_i}, b_{i_j j_i}\right) \right\|_{i\in I}^{j\in J} \right\}_{\mathbf{i}\in\mathbf{I}}^{\mathbf{j}\in\mathbf{J}} \right\rangle. \qquad (1.2.1)$$

1.3 The associated bayesian game

According to [4] and [5] we can construct the bimatrix Bayesian game for the bimatrix incomplete information game $\widetilde{\Gamma}$ from (1.2.1) that consists of the following elements described in [6].

A set of players is $\{1, 2\}$. A set of possible actions for each player: for player 1 is $I = \{1, 2, .., n\}$, the line index, and for player 2 is $J = \{1, 2, .., m\}$, the column index of the payoff matrices.

A set of possible types for each player: the types of the player 1 are $\Delta_1 = \{\alpha = 1, ..., n^m\}$ and of the player 2 are $\Delta_2 = \{\beta = 1, ..., m^n\}$. Only player 1 (player 2) knows his type α (his type β) when play begins. So we say that the player 1 is of the type α if he chooses the matrix $A(\alpha, \beta) \equiv A(\mathbf{i}^\alpha, \mathbf{j}^\beta) = ||a_{i_j^\alpha j_i^\beta}||_{i\in I}^{j\in J}$ as the payoffs and respectively, we say that the player 2 is of the type β if he chooses the matrix $B(\alpha, \beta)$ $\equiv B(\mathbf{i}^\alpha, \mathbf{j}^\beta) = ||b_{i_j^\alpha j_i^\beta}||_{i\in I}^{j\in J}$ as the payoffs. Here $\mathbf{i}^\alpha \in \mathbf{I}$, $\mathbf{j}^\beta \in \mathbf{J}$.

The probability p (respectively q) summarizes what player 1 (respectively player 2), given his type, believes about the types of the other players. So, $p(\beta|\alpha) = \dfrac{p(\beta \cap \alpha)}{p(\alpha)}$ (Bayes'Rule) (respectively

$q(\alpha|\beta) = \dfrac{q(\alpha \cap \beta)}{q(\beta)}$) is the conditional probability assigned to the type $\beta \in \Delta_2$ (respectively $\alpha \in \Delta_1$) when the type of the player 1 is α (respectively of the player 2 is β).

Combining actions and types for each player it is possible to construct the strategies that may assign different actions to different types. In this way we will construct the strategies of the players. If player 1 is of type $\alpha \in \Delta_1$ and he knows that the type of the player 2 may be an element from the set $\Delta_2 = \{\beta = 1, ..., \varkappa_2\}$, and as the utility matrix elements also depend on the type β of player 2, then the set of matrices that represents his utility is $\left\{ A(\alpha, \beta) = ||a_{i_j^\alpha j_i^\beta}||_{i\in I}^{j\in J} \equiv \left\| a_{ij}^{\alpha\beta} \right\|_{i\in I}^{j\in J} \right\}_{\alpha=\overline{1,\varkappa_1}}^{\beta=\overline{1,\varkappa_2}}$. For α-type player 1 we will denote the pure strategy by $\mathbf{l} = l_1 l_2 ... l_\beta ... l_{\varkappa_2}$ and it has the following meaning: the player 1 will choose the line $l_1 \in I$ from the utility matrix $A(\alpha, 1)$ if $\beta = 1$, and line $l_2 \in I$ from the utility matrix $A(\alpha, 2)$ if $\beta = 2$ and so on: line $l_{\varkappa_2} \in I$ from the utility matrix $A(\alpha, \varkappa_2)$ if $\beta = \varkappa_2$. Denote by $\mathbf{L}(\alpha)$ the set of all pure strategy of the α-type player 1. Similarly, if player 2 is of type $\beta \in \Delta_2$ and he knows that the type of player 1 may be an element from the set $\Delta_1 = \{\alpha = 1, ..., \varkappa_1\}$, and as the utility matrix elements depend also on the type α of player 1, then the set of matrices that represents his utility is $\left\{ B(\alpha, \beta) = ||b_{i_j^\alpha j_i^\beta}||_{i\in I}^{j\in J} \equiv \left\| b_{ij}^{\alpha\beta} \right\|_{i\in I}^{j\in J} \right\}_{\alpha=\overline{1,\varkappa_1}}$. So for β-type player 2 we will denote the pure strategy by $\mathbf{c} = c_1 c_2 ... c_\alpha ... c_{\varkappa_1}$ and it has the following meaning: the player will chose column $c_1 \in J$ from utility matrix $B(1, \beta)$ if $\alpha = 1$, and column $c_2 \in J$ from utility matrix $B(2, \beta)$ if $\alpha = 2$ and so on he will chose column $c_{\varkappa_1} \in J$ from utility matrix $B(\kappa_1, \beta)$ if $\alpha = \kappa_1$. Denote by $\mathbf{C}(\beta)$ the set of all pure strategies of β-type player 2. So let the type of the player 1 is α with probability $p = 1$ and the type of the player 2 is β with the probability $q = 1$. Then the payoffs of the player 1 in the strategy profile $(\mathbf{l}, \mathbf{c})$ is the element $a_{ij}^{\alpha\beta}$, where $i = l_\beta$ (the line i from the utility matrix $A(\alpha, \beta)$), $j = c_\alpha$ (the column j from utility matrix $A(\alpha, \beta)$).

A payoff function specifies each player's expected payoff matrices

for every possible combination of all player's actions and types. Hence, if the player 1 is of type α, that is, he will choose the strategy $\mathbf{l}^{\alpha}$ only from the set $\mathbf{L}(\alpha)$, which ensures the payoffs determined by the matrices $A(\alpha,\beta)$, and believes with the probability $p(\beta|\alpha)$ that the player 2 plays the strategy $\mathbf{c}\in\mathbf{C}(\beta)$ for $\beta\in\Delta_2$, then expected payoffs of α-type player 1 is the following matrix

$$\mathbf{A}(\alpha)=\|\mathbf{a_{lc}}\|_{\mathbf{l}\in\mathbf{L}(\alpha)}^{\mathbf{c}\in\mathbf{C}(\beta)} \tag{1.3.1}$$

and

$$\mathbf{a_{lc}}\equiv\mathbf{a}_{l_1l_2\ldots l_\beta\ldots l_{\varkappa_2}\,c_1c_2\ldots c_\alpha\ldots c_{\varkappa_1}}=\sum_{\beta\in\Delta_2}p(\beta/\alpha)a_{l_\beta c_\alpha}\equiv\sum_{\beta\in\Delta_2}p(\beta/\alpha)a_{ij}^{\alpha\beta}.$$

Similarly, if the player 2 is of type β, that is, he will choose the strategy $\mathbf{c}^{\beta}$ only from the set $\mathbf{C}(\beta)$, which ensures the payoffs determined by the matrices $B(\alpha,\beta)$, and believes with the probability $q(\alpha|\beta)$ that the player 1 plays the strategy $\mathbf{l}\in\mathbf{L}(\alpha)$ for $\alpha\in\Delta_1$, then expected payoffs of β-type player 2 is the following matrix

$$\mathbf{B}(\beta)=\|\mathbf{b_{lc}}\|_{\mathbf{l}\in\mathbf{L}(\alpha)}^{\mathbf{c}\in\mathbf{C}(\beta)} \tag{1.3.2}$$

and $\mathbf{b_{lc}}\equiv\mathbf{b}_{l_1l_2\ldots l_\beta\ldots l_{\varkappa_2}\,c_1c_2\ldots c_\alpha\ldots c_{\varkappa_1}}=\sum\limits_{\alpha\in\Delta_1}q(\alpha|\beta)b_{l_\beta c_\alpha}\equiv\sum\limits_{\alpha\in\Delta_1}q(\alpha|\beta)b_{ij}^{\alpha\beta}$. Here $i=l_\beta$ and $j=c_\alpha$. So for the incomplete information game $\widetilde{\Gamma}$ from (1.2.1), the normal form game

$$\Gamma_{Bayes}=\langle\{1,2\},\{\Delta_1,\Delta_2\},\mathbf{L},\mathbf{C},\mathcal{A},\mathcal{B}\rangle, \tag{1.3.3}$$

where $\mathbf{L}=\bigcup\limits_{\alpha\in\Delta_1}\mathbf{L}(\alpha)$, $\mathbf{C}=\bigcup\limits_{\beta\in\Delta_2}\mathbf{C}(\alpha)$ and the utility matrices are $\mathcal{A}=\|\mathbf{A}(\alpha)\|_{\alpha\in\Delta_1}$ and $\mathcal{B}=\|\mathbf{B}(\beta)\|_{\beta\in\Delta_2}$, is called the *associated Bayesian game* in the non informational extended strategies. The matices $\mathcal{A}$ and $\mathcal{B}$ are the "big matrices", that consist of the submatrices of the type $\mathbf{A}(\alpha)$ and $\mathbf{B}(\beta)$, respectively.

The games defined above are sometimes called *Bayesian normal form games*, since the drawing of types is followed by a simultaneous

move game. Denote by $BE[\Gamma_{Bayes}]$ the set of all Bayes-Nash strategies profile of the game Γ_{Bayes} from (1.3.3). For any fixed $\alpha \in \Delta_1$ and $\beta \in \Delta_2$ the game $sub\Gamma_{Bayes} = \langle \{1,2\}, \mathbf{L}(\alpha), \mathbf{C}(\beta), \mathbf{A}(\alpha), \mathbf{B}(\beta) \rangle$ will be called a *subgame of the Bayesian game* Γ_{Bayes} from (1.3.3), and using the notion of "type-players", the $sub\Gamma_{Bayes}$ is the bimatrix game of the α-type player 1 and of the β-type player 2.

So we have the following diagram

$\Gamma \quad \Rightarrow$	$Game\,(1 \leftrightarrows 2) \quad \Rightarrow$	$\widetilde{\Gamma} \quad \Rightarrow$	Γ_{Bayes}
The game in complete and perfect information over the sets of pure strategies	The game in imperfect information over the sets of informational extended strategies	The incomplete and imperfect information game generated by the informational extended strategies	Associated Bayesian game in the non informational extended strategies

Consider the following bimatrix game $H_1 = \begin{pmatrix} 3 & 5 & 4 \\ 6 & 7 & 2 \end{pmatrix}$, $H_2 = \begin{pmatrix} 0 & 5 & 1 \\ 4 & 3 & 2 \end{pmatrix}$ in complete and $(1 \leftrightarrows 2)-$ perfect information and suppose that the informational extended strategies of the player 1 are $\mathbf{i}^1 \equiv i_1 i_2 i_3 = 1_1 1_2 2_3$, $\mathbf{i}^2 \equiv i_1 i_2 i_3 = 1_1 2_2 1_3$ and respectively, of the player 2 are $\mathbf{j}^1 \equiv j_1 j_2 = 1_1 2_2$ and $\mathbf{j}^2 \equiv j_1 j_2 = 2_1 1_2$. The informational extended strategies $\left\{\mathbf{i}^1, \mathbf{i}^2, \mathbf{j}^1, \mathbf{j}^2\right\}$ generate an incomplete information bimatrix game $\widetilde{\Gamma}$ in which the payoff matrix may be one of the following matrices and imperfect information over the set of informational non extended strategies I, J**:**

$$AB\left(\mathbf{i}^1, \mathbf{j}^1\right) = \begin{pmatrix} (3,0)\,(3,0)\,(6,4) \\ (5,5)\,(5,5)\,(7,3) \end{pmatrix}, AB\left(\mathbf{i}^2, \mathbf{j}^1\right) = \begin{pmatrix} (3,0)\,(6,4)\,(3,0) \\ (5,5)\,(7,3)\,(5,5) \end{pmatrix},$$

$$AB\left(\mathbf{i}^1, \mathbf{j}^2\right) = \begin{pmatrix} (5,5)\,(5,5)\,(7,3) \\ (3,0)\,(3,0)\,(6,4) \end{pmatrix}, AB\left(\mathbf{i}^2, \mathbf{j}^2\right) = \begin{pmatrix} (5,5)\,(7,3)\,(5,5) \\ (3,0)\,(6,4)\,(3,0) \end{pmatrix}.$$

The set of strategies of the α-type player 1 $(\alpha = 1)$ is $\mathbf{L}(\alpha = 1) = \{1_1 1_2, 1_1 2_2, 2_1 1_2, 2_1 2_2)\}$ and of the β-type player 2 $(\beta = 1)$ is $\mathbf{C}(\beta = 1) = \{1_1 1_2, 1_1 2_2, 1_1 3_2, 2_1 1_2, 2_1 2_2, 2_1 3_2, 3_1 1_2, 3_1 2_2, 3_1 3_2\}$. Thus a $sub\Gamma_{Bayes} = \langle \{1,2\}, \mathbf{L}(\alpha), \mathbf{C}(\beta), \mathbf{A}(\alpha), \mathbf{B}(\beta) \rangle$ for $\alpha = 1$ and $\beta = 1$ and the believer probability $p = 1/2$, $q = 1/2$ is the following:

$$\mathbf{A} = \begin{pmatrix} -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0.5 & 0.3 & 0.5 \end{pmatrix},$$

$$\mathbf{B} = \begin{pmatrix} 0 & -2 & 0 & 0 & -2 & 0 & 0 & -2 & 0 \\ 0 & -2 & 0 & 0 & -2 & 0 & 0 & -2 & 0 \\ 0 & 1 & 0 & 2 & 1 & 2 & -1 & 0 & -1 \\ 0 & 1 & 0 & 2 & 1 & 2 & -1 & 0 & -1 \end{pmatrix}.$$

Similarly, we can construct the all $sub\Gamma_{Bayes}$.

Finally for determining the Bayes-Nash solutions in the bimatrix informational extended games we use the following theorem [3]:

Theorem 1.1. *The strategy profile* $(\mathbf{l}^*, \mathbf{c}^*)$*, where* $\mathbf{l}^* \in \mathbf{L}$*,* $\mathbf{c}^* \in \mathbf{C}$*, is a Bayes-Nash equilibrium in the game* Γ_{Bayes} *from (1.3.3) if and only if the strategy profile* $(\mathbf{l}^*, \mathbf{c}^*)$ *is a Nash equilibrium for the subgame* $sub\Gamma_{Bayes} = \langle \{1, 2\}, \mathbf{L}(\alpha), \mathbf{C}(\beta), \mathbf{A}(\alpha), \mathbf{B}(\beta) \rangle$*. To determine all Bayes-Nash equilibrium profiles we can determine the all Nash equilibrium profiles for all bimatrix games of type* $sub\Gamma_{Bayes}$ *in the non extended strategies.*

2 Parallel algorithm for mixed system with shared and distributed memory to determine the Bayes-Nash solutions to the bimatrix informational extended games

2.1 Parallel algorithm

The basic parallel strategy consists of three main steps. The **first step** is to partition the input into several partitions of almost equal sizes. The **second step** is to solve recursively the subproblem defined by each partition of the input. Note that these subproblems can be solved concurrently in the parallel system. The **third step** is to combine or merge the solutions of the different subproblems into a solution for the

overall problem. The success of such strategy depends on whether or not we can perform the first and third steps efficiently [7]. To realize the first step of the parallel strategy, that is to realize data paralelization, we use the MPI programming model and open source Scalable Linear Algebra PACKage (ScaLAPACK) or more exactly, the Basic Linear Algebra Subprograms or BLAS routines [8]. The BLACS are a message-passing library designed for linear algebra. The computational model consists of a one- or two-dimensional process grid, where each process stores pieces of the matrices of the $sub\Gamma_{Bayes}$.

So, using the MPI-OpenMP programming model and ScaLAPACK -BLACS packages we can construct the following parallel algorithm to find the all equilibrium profiles $(\mathbf{l}^*, \mathbf{c}^*)$ in the game Γ_{Bayes}.

Algorithm 2.1

1. Using the MPI programming model we generate the virtual medium of MPI-process communication (MPI Communicator) with linear topology and dimension $\varkappa_1 \cdot \varkappa_2$. Root process, using the `MPI_Bcast` function, broadcasts to all MPI process the initial matrices $A = ||a_{ij}||_{i\in I}^{j\in J}$, and $B = ||b_{ij}||_{i\in I}^{j\in J}$ of the bimatrix game $\Gamma = \langle A, B\rangle$.

2. Using the MPI programming model and open source library ScaLAPACK-BLACS, the processes grid $\{(\alpha,\beta)\}_{\alpha=\overline{1,\varkappa_1}}^{\beta=\overline{1,\varkappa_2}}$ is initialized and in parallel (concurrent) all fixed $(\alpha,\beta)-$processes, using combinatorial algorithm, construct the informational extended strategies $\mathbf{i}^\alpha = i_1^\alpha i_2^\alpha ... i_j^\alpha ... i_m^\alpha$ and $\mathbf{j}^\beta = j_1^\beta j_2^\beta ... j_i^\beta ... j_n^\beta$. Thus, the MPI $(\alpha,\beta)-$process has built up the informational extended strategies $\mathbf{i}^\alpha = i_1^\alpha i_2^\alpha ... i_j^\alpha ... i_m^\alpha$ and $\mathbf{j}^\beta = j_1^\beta j_2^\beta ... j_i^\beta ... j_n^\beta$.

3. In parallel, all fixed MPI (α,β)-processes, using the OpenMP directives, construct utility matrices $A(\alpha,\beta) = ||a_{i_j^\alpha j_i^\beta}||_{i\in I}^{j\in J}$ and $B(\alpha,\beta) = ||b_{i_j^\alpha j_i^\beta}||_{i\in I}^{j\in J}$, generated by the informational extended strategies $\mathbf{i}^\alpha$ and $\mathbf{j}^\beta$.

4. In parallel, the MPI α-rank process, for all $\alpha = \overline{1, \varkappa_1}$, generates the "believer-probabilities" $p(\beta/\alpha)$ for all fixed $\beta = \overline{1, \varkappa_2}$ of the α-type player 1 and also, MPI β-rank process, for all $\beta = \overline{1, \varkappa_2}$, generates the "believer-probabilities" $q(\alpha/\beta)$ for all fixed $\alpha = \overline{1, \varkappa_1}$ of the β-type player 2. Thus, the MPI (α, β)-process "possesses" the probabilities $p(\beta/\alpha)$ and $q(\alpha/\beta)$.

5. Using MPI and OpenMP programming models**,** for all $\alpha = \overline{1, \varkappa_1}$, in parallel, the MPI α-rank process generates the sets $\mathbf{L}(\alpha)$ of the $\mathbf{l}^{\alpha} = l_1^{\alpha} l_2^{\alpha} ... l_{\beta}^{\alpha} ... l_{\varkappa_2}^{\alpha}$ strategies and constructs the payoff matrix $\mathbf{A}(\alpha)$ from (1.3.1) of the α-type player 1 and the MPI β-rank process, for all $\beta = \overline{1, \varkappa_2}$, generates the sets $\mathbf{C}(\beta)$ of the $\mathbf{c}^{\beta} = c_1^{\beta} c_2^{\beta} ... c_{\alpha}^{\beta} ... c_{\varkappa_1}^{\beta}$ strategies and constructs the payoff matrix $\mathbf{B}(\beta)$ from (1.3.2) of the β-type player 2. So all MPI (α, β)-processes have a pair of matrices $(\mathbf{A}(\alpha), \mathbf{B}(\beta))$.

6. Denote by $\mathcal{I}$ and $\mathcal{J}$ the set of line and column indices of the matrices $\mathbf{A}(\alpha)$ and $\mathbf{B}(\beta)$. In parallel, all MPI (α, β)-processes, using the OpenMP directives, eliminate from matrix $\mathbf{A}(\alpha)$ and from matrix $\mathbf{B}(\beta)$ the lines that are strictly dominated in matrix $\mathbf{A}(\alpha)$ and columns that are strictly dominated in matrix $\mathbf{B}(\beta)$. Finally we obtain the matrices $\left(\widehat{\mathbf{A}}(\alpha), \widehat{\mathbf{B}}(\beta)\right)$, where $\widehat{\mathbf{A}}(\alpha) = ||\widehat{a}_{ij}||_{i \in \widehat{\mathcal{I}}}^{j \in \widehat{\mathcal{J}}}$ and $\widehat{\mathbf{B}}(\beta) = ||\widehat{b}_{ij}||_{i \in \widehat{I}}^{j \in \widehat{\mathcal{J}}}$, for all $i \in \widehat{\mathcal{I}}$, $j \in \widehat{\mathcal{J}}$ and cardinals $\left|\widehat{\mathcal{I}}\right| \leq |\mathcal{I}|$, $\left|\widehat{\mathcal{J}}\right| \leq |\mathcal{J}|$.

7. In parallel, all MPI (α, β)-processes, using the OpenMP functions, ScaLAPACK routines and the existing sequential algorithm, determine all Nash equilibrium profiles in the bimatrix game with matrices $\left(\widehat{\mathbf{A}}(\alpha), \widehat{\mathbf{B}}(\beta)\right)$ and construct the set of Nash equilibrium profiles in the bimatrix game with matrices $(\mathbf{A}(\alpha), \mathbf{B}(\beta))$.

8. Using ScaLAPACK-BLACS routines, the root MPI process gather from processes grid $\{(\alpha, \beta)\}_{\alpha = \overline{1, \varkappa_1}}^{\beta = \overline{1, \varkappa_2}}$ the sets of Nash equilibrium profiles in the bimatrix game $(\mathbf{A}(\alpha), \mathbf{B}(\beta))$.

In the general case, to determine all sets of Bayes-Nash equilibrium profiles in bimatrix informational extended games a very large number (equal to $n^m \times m^n$) of the $sub\Gamma_{Bayes}$ bimatrix games in the non extended strategies are to be solved. Therefore, it is recommended to use the exascale HPC systems. C++ program using MPI functions, OpenMP directives and ScaLAPACK routines has been developed for this algorithm. Program has been tested on the control examples. The test results were consistent with theoretical results.

2.2 Communication and Run Complexity for Clusters: overview

Parallel program evaluation must take into account the computing system architecture, which means that it is possible that the chosen algorithm is the best for a particular machine but for machines of other architecture it may be not the best parallel algorithm for solving that problem. Also, it is possible, that for a different input size different parallel algorithms to be good for solving the same problem. That is why for comparing the parallel and the sequential variant of an algorithm there must be specified the parallel computing model, must be chosen the best sequential algorithm and must be pointed if there are specific conditions for algorithm performance depending on the input size. In general, in a study of performance of algorithms the following factors are taken into consideration [9]:

- arithmetic operations;
- data transfer.

Estimation of Communication Complexity for Clusters. The time necessary for transmitting data between the processors defines the communication overhead of the duration of parallel algorithm execution in a multiprocessor computer system. The basic set of parameters, which can help to evaluate the data transmission time, consists of the following values:

- initializing time (t_s) characterizes the duration of preparing the message for transmission, the search of the route in the network etc.

- control data transmission time (t_h) between two neighboring processors (i.e. the processors, connected by a physical data transmission channel); to control data we may refer the message header, the error detection data block etc.;
- transmission time of one data byte along a data transmission channel (t_b); the duration of this transmission is defined by the communication channel bandwidth.

If we choose for the further analysis the clusters of this widely used type (the complete graph topology, packet communication method), then the time complexity of the communication operation between two processors may be estimated according to the following formula [10]: $T_{comm}(m) = t_s + m * t_b + t_h$, the estimation of this type is caused by the expression of packet communication method, when the path length of data transmission is $l = 1$. Such an approach is quite possible. However, it is possible to notice that in this model the time of data preparation t_s is assumed to be constant (it does not depend on the amount of the transmitted data). The time of control data transmission t_h does not depend on the number of the transmitted packets, etc. These assumptions do not fully coincide with the real situation, and the time estimations obtained with the help of this model may be not accurate enough.

Time Complexity. The main reason behind developing parallel algorithms was to reduce the computation time of an algorithm. Thus, evaluating the execution time of an algorithm is extremely important in analyzing its efficiency.

Execution time is measured on the basis of the time taken by the algorithm to solve a problem. The total execution time is calculated from the moment when the algorithm starts executing to the moment it stops. If all the processors do not start or end execution at the same time, then the total execution time of the algorithm is the moment when the first processor started its execution to the moment when the last processor stops its execution.

Parallel Random Access Machines (PRAM) is a model, which is considered for most of the parallel algorithms. Here, multiple processors are attached to a single block of memory. A PRAM model contains:

a set of similar type of processors; all the processors share a common memory unit; processors can communicate among themselves through the shared memory only; a memory access unit (MAU) connects the processors with the single shared memory. There are many methods to implement the PRAM model, but the most prominent ones are: shared memory model; message passing model, data parallel model.

In the general case the execution time $T_p(n) = T_{comput} + T_{comm}$, where T_{comput} denotes the computation time, T_{comm} denotes the communication (transferring data) time and n denotes the volume of data.

2.3 Time performance analysis of the parallel algorithm to determine the Bayes-Nash solutions to the bimatrix informational extended games

In this paragraph we will determine some of the numerical characteristics of the parallel Algorithm 2.1 constructed above. For convenience, we use the following notation: $T_p^k(n)$ denotes the time run complexity of parallel implementation of the iteration k of the Algorithm 2.1.

We can demonstrate the following theorem that estimates the run time performance and communication complexity of the parallel Algorithm 2.1.

Theorem. *The run time complexity of the Algorithm 2.1 is*

$$T_{comput} = \sum_{k=2}^{7} T_p^k = O(\max(n,m)) + O\left(\max(\varkappa_1, \varkappa_2)\right) +$$

$$+O\left(\max\left(|\mathcal{I}|, |\mathcal{J}|\right)\right) + O\left(\max\left(\left|\widehat{\mathcal{I}}\right|, \left|\widehat{\mathcal{J}}\right|, |grBr_1| \cdot |grBr_2|\right)\right)$$

and communication complexity is

$$T_{comm} = O\left(t_s + \left[\max\left(|\mathcal{I}| \times |\mathcal{J}|, m \times n\right)\right] * t_b + t_h\right).$$

Proof. For this theorem demonstration it is sufficient to estimate the run time performance and communication complexity of each step of parallel Algorithm 2.1.

Communication complexity of the step 1 of the Algorithm 2.1. If we take into account all considerations from the communications for clusters, then the communication complexity is $T_{comm}^1(m \times n) = t_s +$

$(m \times n) * t_b + t_h$, where $m \times n$ is the dimension of the payoff matrices A and B.

Run time performance of the step 2 of the Algorithm 2.1. The construction of the information extended strategy $\mathbf{i}^\alpha$ is equivalent to the following problem: for given non-negative integers $\{1, 2, ..., i, ...n\}$, to generate a length m string of these numbers. For example, if $I = \{1, 2\}$ and $J = \{1, 2, 3\}$, then the sets of the informational extended strategies are

$$\mathbf{I} = \{1_1 1_2 1_3, 2_1 2_2 2_3, 1_1 1_2 2_3, 1_1 2_2 1_3, 2_1 1_2 1_3, 1_1 2_2 2_3, 2_1 1_2 2_3, 2_1 2_2 1_3\},$$

$$\mathbf{J} = \{1_1 1_2, 2_1 2_2, 3_1 3_2, 1_1 2_2, 2_1 1_2, 1_1 3_2, 3_1 1_2, 2_1 3_2, 3_1 2_2\}.$$

So, to construct the set $\mathbf{I}$ for $n = 2$, $m = 3$, we must: a) generate the strings $(1, 1, 1)$ and $(2, 2, 2)$; b) having the numbers $\{1, 2\}$ to generate all the sub-strings of length 3 with the elements in this set, that are the strings $(1, 1, 2)$, $(1, 2, 1)$, $(2, 1, 1)$, $(1, 2, 2,)$, $(2, 1, 2)$, $(2, 2, 1)$. In mathematics, a multiset (or bag) is a generalization of the concept of a set that, unlike a set, allows multiple instances of the multiset's elements. For example, $\{a, a, b\}$ and $\{a, b\}$ are different multisets although they are the same set. However, order is important, so $\{a, a, b\}$ and $\{a, b, a\}$ are the different multisets. It can be easily noticed that any informational extended strategy is nothing more than a multiset, so their generation actually consists in generating multisets [11]. So we will get that each (α, β)-process generates in $O(m)$ time the strategy $\mathbf{i}^\alpha$ and in $O(n)$ time the strategy $\mathbf{j}^\beta$ using the combinatorial algorithm for generating the multiset permutations. To generate the sets $\mathbf{I}$ and $\mathbf{J}$ of the informational extended strategies the run time complexity is $T_p^2 = O(\max(n, m))$ in the case of using $\varkappa_1 \times \varkappa_2$ processes with distributed memory.

Run time performance of the step 3 of the Algorithm 2.1. This step can be detailed as follows. a) For all (i, j) determine $i_j^\alpha \in I$ and $j_i^\beta \in J$; This operation, using lists, can be performed in $O(1)$ time. b) The assignment operations $a_{ij}^{\alpha\beta} = a_{i_j^\alpha j_i^\beta}$ and $b_{ij}^{\alpha\beta} = b_{i_j^\alpha j_i^\beta}$, the time complexity of which is $O(n \cdot m)$ (the number of elements in matrices $A(\alpha, \beta)$, respectively $B(\alpha, \beta)$). So, if for each (α, β)-process

$n \cdot m$ OpenMP processes (threads) are generated (that is, (α, β) -process "becomes" a parallel system with a PRAM computation model), then the elements $a_{ij}^{\alpha\beta}$, $b_{ij}^{\alpha\beta}$ are generated in parallel and the time complexity $T_p^3(n \cdot m) = O(1)$, using $n \cdot m$ parallels shared memory (SMM) processes.

Run time performance of the step 4 of the Algorithm 2.1. The α-process will generate "believer-probabilities" $p(\beta/\alpha)$ for all $\beta = \overline{1, \varkappa_2}$ in $O(\varkappa_2)$ time. If for each α-process $\varkappa_2$ OpenMP process (thread) is generated (that is, α-process "becomes" a parallel system with a PRAM computation model), then probabilities $p(\beta/\alpha)$ are generated in parallel (each thread generates $p(\beta/\alpha)$ for a fixed β) in $O(1)$ time. Similarly, β-process will generate "believer-probabilities" $q(\alpha/\beta)$ for all $\alpha = \overline{1, \varkappa_1}$ in $O(\varkappa_1)$ time. If for each β-process $\varkappa_1$ OpenMP process (thread) is generated (that is, β-process "becomes" a parallel system with a PRAM computation model), then probabilities $q(\alpha/\beta)$ are generated in parallel (each thread generates $q(\alpha/\beta)$ for a fixed α) in $O(1)$ time. So, in using $\varkappa_1 \times \varkappa_2$ parallel distribute memory (DMM) processes, the time complexity is $T_p^4 = O(1)$, and for each of these processes it is generated $max(\varkappa_1, \varkappa_2)$ threads (that are SMM type processes).

Run time performance of the step 5 of the Algorithm 2.1. This step can be divided into the next two.
5.1. Time complexity for generating the sets $\mathbf{L}(\alpha)$ and $\mathbf{C}(\beta)$). Similarly to step 2, process α in $O(\varkappa_2)$ time generates the sets $\mathbf{L}(\alpha)$ of the $\mathbf{l}^\alpha = l_1^\alpha l_2^\alpha ... l_\beta^\alpha ... l_{\varkappa_2}^\alpha$ strategies and process β in $O(\varkappa_1)$ time generates the sets $\mathbf{C}(\beta)$ of the $\mathbf{c}^\beta = c_1^\beta c_2^\beta ... c_\alpha^\beta ... c_{\varkappa_1}^\beta$ strategies using the combinatorial algorithm for generating the multiset permutations. Here for every fixed α we have $\varkappa_2$ strategies of the type $\mathbf{l}^\alpha = l_1^\alpha l_2^\alpha ... l_\beta^\alpha ... l_{\varkappa_2}^\alpha$ (the number of the type of player 2) and for all fixed β we have $\varkappa_1$ strategies of the type $\mathbf{c}^\beta = c_1^\beta c_2^\beta ... c_\alpha^\beta ... c_{\varkappa_1}^\beta$ (the number of the type of player 1). If for each α-process $\varkappa_2$, the OpenMP process (thread) is generated (that is, α-process "becomes" a parallel system with a PRAM computation model), then the set $\mathbf{L}(\alpha)$ is generated in parallel (each thread produces $\mathbf{l}^\alpha = l_1^\alpha l_2^\alpha ... l_\beta^\alpha ... l_{\varkappa_2}^\alpha$ for all fixed β) in $O(1)$ time. Similarly, if for each β-process, the $\varkappa_1$ OpenMP process (thread) is generated (that is β-process "becomes" a parallel system with a PRAM compu-

tation model), then the set $\mathbf{C}(\beta)$ is generated in parallel (each thread produces $\mathbf{c}^{\beta} = c_1^{\beta} c_2^{\beta} ... c_{\alpha}^{\beta} ... c_{\varkappa_1}^{\beta}$ for all fixed α) in $O(1)$ time. So, for all (α, β)-MPI processes, time complexity for construction the sets $\mathbf{L}(\alpha)$ and $\mathbf{C}(\beta)$ is $O(1)$ in case of using $max(\varkappa_1, \varkappa_2)$ threads (that are SMM type processes).
5.2. Time complexity for generating the matrices $\mathbf{A}(\alpha), \mathbf{B}(\beta)$, that is the elements $\mathbf{a}_{\mathbf{l}^{\alpha}\mathbf{c}^{\beta}}$ and $\mathbf{b}_{\mathbf{l}^{\alpha}\mathbf{c}^{\beta}}$. Time complexity for determination of the element $\mathbf{a}_{\mathbf{l}^{\alpha}\mathbf{c}^{\beta}}$ is $O(\varkappa_2)$ (maximum number of iterations), and similarly for element $\mathbf{b}_{\mathbf{l}^{\alpha}\mathbf{c}^{\beta}}$, it is $O(\varkappa_1)$. If for all (α, β)-MPI processes there are generated $\varkappa_1 \times \varkappa_2$ threads (how many elements must be constructed, i.e. the number of matrix lines and columns of the matrices $\mathbf{A}(\alpha), \mathbf{B}(\beta)$), then the time complexity for generating the matrices $\mathbf{A}(\alpha), \mathbf{B}(\beta)$ is $O(\max(\varkappa_1, \varkappa_2))$.
Finally, if for step 5 we use $\max(\varkappa_1, \varkappa_2)$ MPI processes and for each of these processes we generate $\varkappa_1 \times \varkappa_2$ execution threads, then the complexity time $T_p^5 = O(\max(\varkappa_1, \varkappa_2))$.

Run time performance of the step 6 of the Algorithm 2.1. So, consider a (α, β)-MPI process, which will "build" the $\left(\widehat{\mathbf{A}}(\alpha), \widehat{\mathbf{B}}(\beta)\right)$ matrix pair. The parallel algorithm can be organized as follows: for these MPI processes we generate $\max\left(\frac{|\mathcal{I}|}{2}, \frac{|\mathcal{J}|}{2}\right)$ OpenMP threads and each thread performs element comparison between two lines of the matrix $\mathbf{A}(\alpha)$ and two columns of the matrix $\mathbf{B}(\beta)$ to determine dominant lines and columns. The complexity of these operations (that is, the comparison of the two elements by two) is $O(\max(|\mathcal{I}|, |\mathcal{J}|))$. Here $|\mathcal{I}|, |\mathcal{J}|$ denotes the number of lines and the number of columns of the matrix $\mathbf{A}(\alpha)$ and $\mathbf{B}(\beta)$ respectively. Thus, if for Step 6 there are used $\max(\varkappa_1, \varkappa_2)$ MPI processes, and for each of these processes there will be generated $\max\left(\frac{|\mathcal{I}|}{2}, \frac{|\mathcal{J}|}{2}\right)$ OpenMP execution threads, then complexity time $T_p^6 = O(\max(|\mathcal{I}|, |\mathcal{J}|))$.

Run time performance of the step 7 of the Algorithm 2.1. So, consider a (α, β)-MPI process, which will determine Nash's equilibrium solutions in pure strategies for bimatrix game $\left(\widehat{\mathbf{A}}(\alpha), \widehat{\mathbf{B}}(\beta)\right)$. The so-

lution is determined based on the definition. Equilibrium profile is the index pair (i^*, j^*), for which the following inequality system is checked: $(i^*, j^*) \Leftrightarrow \begin{cases} \widehat{a}_{i^*j^*} \geq \widehat{a}_{ij^*} \, \forall i \in \widehat{\mathcal{I}} \\ \widehat{b}_{i^*j^*} \geq \widehat{b}_{i^*j} \, \forall j \in \widehat{\mathcal{J}} \end{cases}$. Or equivalently: a) determine $i^*(j) = \arg\max\limits_{i \in \widehat{\mathcal{I}}} \widehat{a}_{ij}$, and $j^*(i) = \arg\max\limits_{j \in \widehat{\mathcal{J}}} \widehat{b}_{ij}$; b) determine those pairs of indices$(i^*, \, j^*)$ for which $i^* = i^*(j^*) \; J^* = j^*(i^*)$. If we use the best response sets of the players, then we will have the following. Let $Br_1(j) = Arg \arg\max\limits_{i \in \widehat{\mathcal{I}}} \widehat{a}_{ij^*}$, $Br_2(i) = Arg \max\limits_{j \in \widehat{\mathcal{J}}} \widehat{b}_{i^*j} \, (i, \, j)$ be the best response set of strategies of the player 1 and respectively of the player 2. Denote by $grBr_1 = \left\{(i,j) : i \in Br_1(j), j \in \widehat{\mathcal{J}}\right\}$ and $grBr_2 = \left\{(i,j) : j \in Br_2(i), i \in \widehat{\mathcal{I}}\right\}$ the graph of these point-to-set mappings. Then $(i^*, \, j^*)$ is Nash equilibrium profile if and only if $(i^*, \, j^*) \in grBr_1 \bigcap grBr_2$. Thus, in order to determine equilibrium profiles in pure strategies for bimatrix games, we must: a) for any fixed column in matrix $\widehat{\mathbf{A}}(\alpha)$ we note (highlight) all the maximum elements per line, that is, we determine $i^*(j) = \arg\max\limits_{i \in \widehat{\mathcal{I}}} \widehat{a}_{ij}$ for all $j \in \widehat{\mathcal{J}}$; b) for any fixed line in matrix $\widehat{\mathbf{B}}(\beta)$ we note (highlight) all the maximum elements per column, that is, we determine $j^*(i) = \arg\max\limits_{j \in \widehat{\mathcal{J}}} \widehat{b}_{ij}$ for all $i \in \widehat{\mathcal{I}}$; c) we select those pairs of indices that are simultaneously highlighted both in matrix $\widehat{\mathbf{A}}(\alpha)$ and matrix $\widehat{\mathbf{B}}(\beta)$. In other words, it is determined $\begin{cases} i^* \equiv i^*(j^*) \\ j^* \equiv j^*(i^*) \end{cases}$, namely $(i^*, \, j^*) \in grBr_1 \bigcap grBr_2$. To determine $i^*(j)$ for a fixed j, the time complexity will be $O(\left|\widehat{\mathcal{I}}\right|)$ and similarly, to determine $j^*(i)$ for fixed i, the time complexity will be $O(\left|\widehat{\mathcal{J}}\right|)$. If for α-MPI process $\left|\widehat{\mathcal{J}}\right|$ threads will be generated and therefore each thread in parallel will determine $i^*(j)$, then to build the set $grBr_1$ time complexity will be $O(\left|\widehat{\mathcal{I}}\right|)$. Similarly, if for β-MPI process $\left|\widehat{\mathcal{I}}\right|$ threads will be generated and each thread in parallel will determine $j^*(i)$, then to build the set $grBr_2$ the time complexity will be

$O(\left|\widehat{\mathcal{J}}\right|)$. Each (α, β) process can sequentially determine $grBr_1 \bigcap grBr_2$ in $O\left(|grBr_1| \cdot |grBr_2|\right)$ time, where $|grBr_1|$ means the number of elements from $grBr_1$. So, if for Step 7 it will be used $\max(\varkappa_1, \varkappa_2)$ MPI processes and for each of these processes $\max\left(\left|\widehat{\mathcal{I}}\right|, \left|\widehat{\mathcal{J}}\right|\right)$ OpenMP threads will be generated, then we obtain the following complexity time $T_p^7 = O\left(\max\left(\left|\widehat{\mathcal{I}}\right|, \left|\widehat{\mathcal{J}}\right|, |grBr_1| \cdot |grBr_2|\right)\right)$.

Communication complexity of the step 8 of the Algorithm 2.1. If we take into account all considerations from the communication for clusters, then the communication complexity is $T_{comm}^8(|\mathcal{I}| \times |\mathcal{J}|) = t_s + (|\mathcal{I}| \times |\mathcal{J}|) * t_b + t_h$, where $|\mathcal{I}| \times |\mathcal{J}|$ is the dimension of the payoff matrices $\mathbf{A}(\alpha)$ and $\mathbf{B}(\beta)$ respectively.

Summarizing the obtained run time and communication complexity of the steps 1)-8) of the *Algorithm 2.1*, we complete the proof of the theorem. ■

3 Concluzion

Information issues are very important for mathematical modeling decision making problems in situations of risk and conflict. In this article, to solve the game in complete and perfect information over the sets of pure strategies, we elaborate the parallel algorithm that in parallel generates the sets of informational extended strategies of the player 1 and player 2; in parallel constructs the incomplete and imperfect information game $\widetilde{\Gamma}$ generated by the informational extended strategies; in parallel constructs the associated Bayesian game Γ_{Bayes} in the non-informational extended strategies. The main complexity here consists in determination of the Bayes-Nash's solutions for the game Γ_{Bayes}, since we have to deal with very large matrices, the elements of which are matrices as well. So, the elaborated algorithm in parallel generates the set of all $sub\Gamma_{Bayes}$ bimatrix subgames in the non-extended strategies, Nash solutions of which are Bayes-Nash solutions. For software implementation on HPC cluster parallel systems, which are in most cases mixed systems with distributed and shared memory, we use the MPI and OpenMP programming models. Also we estimate the run

time performance and communication complexity of the elaborated algorithm. The obtained estimates demonstrate that the effectiveness of the algorithm will increase if the exascale HPC systems will be used.

References

[1] N. S. Kukushkin, V. V. Morozov, *Theory of non-antagonistic games*, Moscow: MSU, 1984 (in Russian).

[2] L. Novac, "Nash equilibria in the noncooperative informational extended games," *Buletinul Academiei de Ştiinţe a Republicii Moldova, Matematica,* no. 1(59), pp. 96–103, 2009.

[3] Boris Hâncu, Mihai Cocîrlă. "Approaches for solving bimatrix informational extended games," *Studia Universitatis Moldaviae, seria Stiinte exacte si economice*, no. 7(87), pp. 71–85, Chisinau 2015.

[4] John C. Harsanyi, "Games with incomplete information played by Bayesian players. Part III: The basic probability distribution of the game," *Management Science*, no. 14, pp. 486–502, 1968.

[5] John C. Harsanyi, Reinhard Selten, *A General Theory of Equilibrium Selection in Games,* Cambridge, MA: MIT-Press, 1998.

[6] Boris Hancu, "Solving two person games with complete and perfect informations," *Studia Universitatis, Seria Ştiinţe exacte şi economice*, no. 7(77), pp. 3–16, Chişinău, 2014.

[7] Joseph Jaja, *An Introduction to Parallel Algorithms*, Addison-Wesley Publishing Company, Inc., 1992.

[8] "ScaLAPACK – Scalable Linear Algebra PACKage". [Online]. Available: http://www.netlib.org/scalapack/.

[9] Virginia Niculescu, *Parallel Calculation. Design and formal development of parallel programs*, Cluj-Napoca: Presa Universitara Clujeana, 2006, 307 p. (in Romanian)

[10] A. Grama, G. Karypis, V. Kumar, A. Gupta, "3. The Estimation of the Communication Complexity of Parallel Algorithms," in *Introduction to Parallel Computing*. The Benjamin/Cummings Publishing Company, Inc. 2nd ed., 2003, ch. 3. [Online]. Available: http://www.hpcc.unn.ru/mskurs/ENG/DOC/pp03.pdf.

[11] Frank Ruskey, "Combinatorial Generation," October 1, 2003. [Online]. Available: http://www.1stworks.com/ref/ruskeycombgen.pdf.

Hancu Boris, Gladei Anatolie

Received September 21, 2017

Institution: Moldova State University
Address: Chisinau, Mateevici str. 60
Phone: 079030211
E–mail: boris.hancu@gmail.com