Proceedings of the International Conference IMCS-60, MSU
October 10–13, 2024, Chisinau, Republic of Moldova

Data Parallelization for Solving Bimatrix Games

Boris Hâncu, Calin Turcanu

Abstract

Contemporary decision-making problems are very complex and require the processing of a very large volume of data. Thus, for the mathematical modelling of these processes, it is necessary to take into account the big data problems. The data is too big to be stored and processed by a single machine. In many large-scale solutions, data is divided into partitions that can be managed and accessed separately. In order to solve such problems in real time, parallel algorithms are built and then implemented on various types of parallel computing systems. In this paper, we will analyze the ways to build parallel algorithms, and especially, data parallelization, for a class of noncooperative games, bimatrix games.

Keywords: bimatrix games, equilibrium profiles, HPC systems, parallel algorithms.

1 Matrix games and algorithms for determining equilibrium profiles in pure strategies.

We consider the bimatrix game in the following strategic form $\Gamma = \langle I, J, A, B\rangle$, 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. So, the game is incomplete and has imperfect information. Players intend to maximize

©2024 by Boris Hâncu, Calin Turcanu

their payoffs. The matrices A and B are called *global matrices*. We denote by $NE[\Gamma]$ the set of all equilibrium profiles in the game Γ.

We will build the following point-to-set application $Br_1 : J \to 2^I$ such that, for any fixed column $j \in J$, $Br_1(j) = Arg\max\limits_{i\in I} a_{ij} = \left\{ i^* \in I : i^* = \arg\max\limits_{i\in I} a_{ij} \right\}$ is determined. Similarly, $Br_2 : I \to 2^J$ such that, for any fixed row $i \in I$, $Br_2(i) = Arg\max\limits_{j\in J} b_{ij} = \left\{ j^* \in J : j^* = \arg\max\limits_{j\in J} b_{ij} \right\}$ is determined. The graphs of Br_1 and Br_2 are denoted by $GrBr_1$ and $GrBr_2$. Then $(i^*,\ j^*) \in NE[\Gamma] \Leftrightarrow (i^*,\ j^*) \in GrBr_1 \bigcap GrBr_2$.

Using the definition, the following *basic sequential algorithm* can be used to determine the Nash equilibrium profiles in pure strategies:

1) for any fixed column $j \in J$, the sets $Br_1(j) = Arg\max\limits_{i\in I} a_{ij}$ and $Br_2(i) = Arg\max\limits_{j\in J} b_{ij}$ are determined;

2) the graphs $GrBr_1$, $GrBr_1$ of the point-to-set application Br_1 and Br_2 are built;

3) construct the set $NE = GrBr_1 \cap GrBr_2$.

2 Generation of sets of bimatrix subgames as a result of the division into blocks of global matrices.

We denote by R the set of processes (processors, computing elements) of a parallel computing system and by I_r and J_r – the lines (columns) of the matrices A_r and B_r, respectively, distributed to the process $r \in R$. For the characterization of the algorithms of division and distribution on matrix-like systems, we will introduce the following functions. The functions $\alpha : I \times R \to I_r \beta : J \times R \to J_r$ will be called Division and Distribution of the Elements of a Matrix (*D&DEM* function), if they verify the following property: for any $i \in I$, there is a single process r and $i_r \in I_r$, such that $\alpha(r, i) = i_r$, and similarly, for any $j \in J$, there is

only one process r and $j_r \in J_r$, so that $\beta(r,j) = j_r$. Also the functions $\varphi : I_r \to I$ and $\psi : J_r \to J$ will be called Restoring the Elements of a Matrix (*REM* function), if they verify the following property: for any $r \in R$ and $i_r \in I_r$, there is a single element $i \in I$ such that $\varphi(i_r) = i$ and similarly, for any $r \in R$ and $j_r \in J_r$, there is a single element $j \in J$, so that $\psi(j_r) = j$. Thus, using the D&DEM functions, we will obtain a series of sub-matrices $\left\{A_r = \left\|a_{\alpha(r,i)\beta(r,j)}\right\|_{i\in I}^{j\in J} \equiv ||a^r_{i_r j_r}||_{i_r\in I_r}^{j_r\in J_r}\right\}_{r\in R}$, and $\left\{B_r = \left\|b_{\alpha(r,i)\beta(r,j)}\right\|_{r\in R} \equiv ||b^r_{i_r j_r}||_{i\in I_r}^{j\in J_r}\right\}_{r\in R}$ which are a result of the division and distribution of matrices A and correspondingly B. Using the REM and submatrices $\{A_r\}_{r\in R}$, $\{B_r\}_{r\in R}$, we will reconstruct the global matrices A and B. If the submatrices have the same size, then these submatrices can generate a series of sub-games $\Gamma_r = \langle I_r, J_r, A_r, B_r\rangle$ and denote by $NE[\Gamma_r]$ the set of Nash equilibrium profiles. Thus, an algorithm for dividing matrices into sub-matrices is uniquely characterized by *D&DEM* and *REM* functions.

For example, if we want to divide the matrices into consecutive n_r lines of the matrix A and m_r columns of the matrix B, the *D&DEM* and *REM* functions will show the following form: D&DEM functions

$$\alpha(r,i) = \begin{cases} i - \sum\limits_{k=0}^{r-1} n_k & \text{for } A, \\ i & \text{for } B, \end{cases}, \quad \beta(r,j) = \begin{cases} j & \text{for } A, \\ j - \sum\limits_{k=0}^{r-1} m_k & \text{for } B, \end{cases}$$

REM functions

$$\varphi(r,i_r) = \begin{cases} i_r + \sum\limits_{k=0}^{r-1} n_k & \text{for } A, \\ i_r & \text{for } B, \end{cases}, \quad \psi(r,j_r) = \begin{cases} j & \text{for } A, \\ j_r + \sum\limits_{k=0}^{r-1} m_k & \text{for } B. \end{cases}$$

3 Basic Parallel Algorithm to find Nash equilibrium profiles

We will present the following parallel algorithm for determining the solutions of matrix games in pure strategies.

A) Data paralelizations: using the *D&DEM* functions, each process $r \in R$ "gets" the pair of matrices A_r, B_r.

B) Each process $r \in R$, independently using the *Basic sequential algorithm*, will construct the following point-to-set applications $Br_1^r(j_r)$ for all $j_r \in J_r$ and $Br_2^r(i_r)$ for all $i_r \in I_r$, and also the set of index pairs $GrBr_1^r \cap GrBr_2^r$ and send these sets to the root process. If, as a result of the division of matrices A and B, subgames $\Gamma_r = \langle I_r, J_r, A_r, B_r \rangle$ are generated, then $NE[\Gamma_r] = GrBr_1^r \cap GrBr_2^r$;

C) Root process, using the sets $\{GrBr_1^r\}_{r \in R}$, $\{GrBr_2^r\}_{r \in R}$ and the *REM* functions, the graph $GrBr_1$ of the point-to-set application Br_1 and also the graph $GrBr_2$ of the point-to-set application Br_2 are built. The equilibrium profiles are all the profiles belonging to the intersection of the two given graphs $NE = GrBr_1 \cap GrBr_2$.

4 Conclusion

This article presents a new way to solve bimatrix games with very large matrices using parallel algorithms. The implementation of parallel algorithms depends a lot on the methods of dividing matrices into sub-matrices and their distribution on an HPC-type computing system, i.e., on achieving data parallelization. For the row and column subarrays, the described algorithms were implemented in software on the USM HPC system.

Boris Hâncu[1], Calin Turcanu[2]

[1]Moldova State University
E–mail: boris.hincu@usm.md
ORCID: https//orcid.org/0009-0008-3519-3419

[2]Moldova State University
E–mail: calin.turcanu@usm.md
ORCID: https://orcid.org/0009-0005-3480-6373