The Search for Systems of Diagonal Latin Squares Using the SAT@home Project

Oleg Zaikin and Stepan Kochemazov

***Abstract*—In this paper we consider one approach to searching for systems of orthogonal diagonal Latin squares. It is based on the reduction of an original problem to the Boolean Satisfiability problem. We describe two different propositional encodings that we use. The first encoding is constructed for finding pairs of orthogonal diagonal Latin squares of order 10. Using this encoding we managed to find 17 previously unknown pairs of such squares using the volunteer computing project SAT@home. The second encoding is constructed for finding pseudotriples of orthogonal diagonal Latin squares of order 10. Using the pairs found with the help of SAT@home and the second encoding we successfully constructed several new pseudotriples of diagonal Latin squares of order 10.**

***Keywords*—Latin squares, Boolean satisfiability problem, volunteer computing, SAT@home.**

I. INTRODUCTION

The combinatorial problems related to Latin squares, which are a form of combinatorial design [1], attract the attention of mathematicians for the last several centuries. In recent years a number of new computational approaches to solving these problems have appeared. For example in [2] it was shown that there is no finite projective plane of order 10. It was done using special algorithms based on constructions and results from the theory of error correcting codes [3]. Corresponding experiment took several years, and on its final stage employed quite a powerful (at that moment) computing cluster. More recent example is the proof of hypothesis about the minimal number of clues in Sudoku [4] where special algorithms were used to enumerate and check all possible Sudoku variants. To solve this problem a modern computing cluster had been working for almost a year. In [5] to search for some sets of Latin squares a special program system based on the algorithms of search for maximal clique in a graph was used.

Also, in application to the problems of search for combinatorial designs, the SAT approach shows high effectiveness [6]. It is based on reducing the original problem to the Boolean satisfiability problem (SAT) [7]. All known SAT solving algorithms are exponential in the worst case since SAT itself is NP-hard. Nevertheless, modern SAT solvers successfully cope with many classes of instances from different areas, such as verification, cryptanalysis, bioinformatics, analysis of collective behavior, etc.

For solving hard SAT instances it is necessary to involve significant amounts of computational resources. That is why the improvement of the effectiveness of SAT solving algorithms, including the development of algorithms that are able to work in parallel and distributed computing environments is a very important direction of research. In 2011 for the purpose of solving hard SAT instances there was launched the volunteer computing project SAT@home [8]. One of the aims of the project is to find new combinatorial designs based on the systems of orthogonal Latin squares.

The paper is organized as follows. In the second section we discuss relevant problems regarding the search for systems of orthogonal Latin squares. In the third section we describe the technique we use to construct the propositional encodings of the considered problems. The fourth section discusses the computational experiment on the search for pairs of orthogonal diagonal Latin squares of order 10 that was held in SAT@home. Later in the same section we show the results obtained for the search of pseudotriples of orthogonal diagonal Latin squares of order 10, using the computing cluster.

II. SOME RELEVANT PROBLEMS OF SEARCH FOR SYSTEMS OF LATIN SQUARES

The Latin square [1] of order n is the square table $n \times n$ that is filled with the elements from some set $M, |M| = n$ in such a way that in each row and each column every element from M appears exactly once. Leonard Euler in his works considered as M the set of Latin letters, and that is how the Latin squares got their name. Hereinafter, M denotes the set $\{0, \dots, n-1\}$.

Two Latin squares A and B of the same order n are called orthogonal if all ordered pairs of the kind $(a_{ij}, b_{ij}), i, j \in \{0, n-1\}$, are different. If there is a set of k different Latin squares among which each two squares are orthogonal, then this set is called a system of k mutually orthogonal Latin squares (MOLS). The question if there exist 3 MOLS of order 10 is of particular interest since this problem remains unanswered for many years. From the computational point of view the problem is very difficult, therefore it is interesting to search for such triples of Latin squares of order 10 for which the orthogonality condition is somehow weakened. For example, we can demand that it should hold in full only for one (two) pairs of squares out of three and only partly for the remaining two (one). There can be other

Manuscript received October 8, 2015. This work was supported in part by the Russian Foundation for Basic Research (grants 14-07-00403-a and 15-07-07891-a) and by the Council on grants of the President of Russian Federation (grants SP-1184.2015.5 and NSH-5007.2014.9).

Oleg Zaikin is a researcher at Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences, e-mail: zaikin.icc@gmail.com.

Stepan Kochemazov is a programmer at Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences; e-mail: veinamond@gmail.com.

variants of weakening this condition. In the remainder of the paper we will refer to such systems of squares as *pseudotriples*.

In this paper we consider the following weakened variant of the orthogonality condition: we fix the number of ordered pairs of elements for which the orthogonality condition should hold simultaneously for all three pairs of squares (A and B, A and C, B and C), comprising the pseudotriple A, B, C. The corresponding number of pairs of elements we call the *characteristics* of the pseudotriple considered. Currently the record pseudotriple in this notation is the one published in [9] (see Fig. 1). In this pseudotriple square A is fully orthogonal to squares B and C, but squares B and C are orthogonal only over 91 pairs of elements out of 100. It means that in our notation the characteristics of this pseudotriple is 91.

$$A=\begin{bmatrix}0&8&9&7&5&6&4&2&3&1\\9&1&4&6&2&7&3&8&0&5\\7&4&2&5&1&3&8&6&9&0\\8&6&5&3&9&2&1&0&4&7\\6&2&1&8&4&0&9&5&7&3\\4&9&3&2&7&5&0&1&6&8\\5&3&7&1&0&8&6&9&2&4\\3&5&0&9&8&4&2&7&1&6\\1&7&6&0&3&9&5&4&8&2\\2&0&8&4&6&1&7&3&5&9\end{bmatrix}\quad B=\begin{bmatrix}0&7&8&9&1&2&3&4&5&6\\9&0&6&1&8&3&2&5&4&7\\7&2&0&4&3&9&1&8&6&5\\8&5&3&0&2&1&7&6&9&4\\6&9&5&3&0&7&4&2&1&8\\4&1&7&6&5&0&8&9&3&2\\5&4&2&8&9&6&0&3&7&1\\1&8&4&2&6&5&9&7&0&3\\2&3&9&5&7&4&6&1&8&0\end{bmatrix}\quad C=\begin{bmatrix}0&7&8&9&1&2&3&4&5&6\\6&4&2&8&9&5&1&3&7&0\\4&9&5&3&2&7&6&0&1&8\\5&1&7&6&4&3&8&9&0&2\\1&0&3&7&6&8&2&5&4&9\\2&8&0&1&3&4&9&7&6&5\\3&2&9&0&7&1&5&8&4&6\\9&5&4&2&8&6&0&1&3&7\\7&3&6&5&0&9&4&8&2&1\\8&6&1&4&5&0&7&2&9&3\end{bmatrix}$$

Fig. 1. Record pseudotriple of order 10 from [9]

In this paper we develop the SAT approach for solving the problem described above. To apply the SAT approach one has to reduce the original problem to the Boolean equation in the form "CNF=1" (here CNF stands for conjunctive normal form). Corresponding transition process is usually referred to as encoding the original problem to SAT. First attempts on the application of the SAT approach to finding systems of orthogonal Latin squares started in the 90-ies years of XX century. A lot of useful information in this area can be found in [6]. In particular, the author of [6] have been trying to find three mutually orthogonal Latin square of order 10 for more than 10 years using a specially constructed grid system of 40 PCs (however, without any success).

In our opinion, it is also interesting to search for systems of orthogonal diagonal Latin squares. The Latin square is called diagonal if both its primary and secondary diagonals contain all numbers from 0 to $n-1$, where n is the order of the Latin square. In other words, the constraint on the uniqueness is extended from rows and columns to two diagonals. The existence of a pair of mutually orthogonal diagonal Latin squares (MODLS) of order 10 was proved only in 1992 – in the paper [10] three such pairs were presented.

Similar to the problem of search for pseudotriples of Latin squares we can consider the problem of search for pseudotriples of diagonal Latin squares of order 10. In available sources we have not found if the problem in such formulation have been studied. Implicitly, however, in [10] one of squares in the first and the second pairs is the same. Figures 2 and 3 depict the corresponding pairs.

Based on the pairs shown in Fig. 2 and Fig. 3 it is easy to construct the pseudotriple of diagonal Latin squares of order 10 (see Fig. 4). The characteristics of this pseudotriple is equal to 60.

In the next section we describe the propositional encodings that we used in our experiments.

$$A=\begin{bmatrix}0&9&4&6&1&7&5&8&2&3\\7&1&9&4&5&3&8&0&6&2\\4&6&2&8&3&1&7&5&9&0\\6&0&7&3&2&8&4&9&1&5\\5&3&6&7&4&2&9&1&0&8\\8&4&1&2&9&5&0&6&3&7\\2&5&3&0&8&9&6&4&7&1\\3&2&8&9&0&4&1&7&5&6\\9&7&5&1&6&0&3&2&8&4\\1&8&0&5&7&6&2&3&4&9\end{bmatrix}\quad B=\begin{bmatrix}0&8&5&1&7&3&4&6&9&2\\5&1&7&2&9&8&0&3&4&6\\1&7&2&9&5&6&8&0&3&4\\9&6&4&3&0&2&7&1&5&8\\3&0&8&6&4&1&5&9&2&7\\4&3&0&8&6&5&9&2&7&1\\7&2&9&5&1&4&6&8&0&3\\6&4&3&0&8&9&2&7&1&5\\2&9&6&4&3&7&1&5&8&0\\8&5&1&7&2&0&3&4&6&9\end{bmatrix}$$

Fig. 2. First pair of MODLS of order 10 from [10]

$$A=\begin{bmatrix}0&4&1&9&8&2&7&3&5&6\\3&1&6&8&2&9&4&5&0&7\\6&5&2&4&9&0&3&8&7&1\\1&8&5&3&7&4&9&0&6&2\\9&2&0&5&4&7&8&6&1&3\\8&6&3&7&1&5&0&9&2&4\\4&0&7&2&5&3&6&1&9&8\\2&9&4&1&6&8&5&7&3&0\\7&3&9&6&0&1&2&4&8&5\\5&7&8&0&3&6&1&2&4&9\end{bmatrix}\quad B=\begin{bmatrix}0&8&5&1&7&3&4&6&9&2\\5&1&7&2&9&8&0&3&4&6\\1&7&2&9&5&6&8&0&3&4\\9&6&4&3&0&2&7&1&5&8\\3&0&8&6&4&1&5&9&2&7\\4&3&0&8&6&5&9&2&7&1\\7&2&9&5&1&4&6&8&0&3\\6&4&3&0&8&9&2&7&1&5\\2&9&6&4&3&7&1&5&8&0\\8&5&1&7&2&0&3&4&6&9\end{bmatrix}$$

Fig. 3. Second pair of MODLS of order 10 from [10]

$$A=\begin{bmatrix}0&8&5&1&7&3&4&6&9&2\\5&1&7&2&9&8&0&3&4&6\\1&7&2&9&5&6&8&0&3&4\\9&6&4&3&0&2&7&1&5&8\\3&0&8&6&4&1&5&9&2&7\\4&3&0&8&6&5&9&2&7&1\\7&2&9&5&1&4&6&8&0&3\\6&4&3&0&8&9&2&7&1&5\\2&9&6&4&3&7&1&5&8&0\\8&5&1&7&2&0&3&4&6&9\end{bmatrix}\quad B=\begin{bmatrix}0&9&4&6&1&7&5&8&2&3\\7&1&9&4&5&3&8&0&6&2\\4&6&2&8&3&1&7&5&9&0\\6&0&7&3&2&8&4&9&1&5\\5&3&6&7&4&2&9&1&0&8\\8&4&1&2&9&5&0&6&3&7\\2&5&3&0&8&9&6&4&7&1\\3&2&8&9&0&4&1&7&5&6\\9&7&5&1&6&0&3&2&8&4\\1&8&0&5&7&6&2&3&4&9\end{bmatrix}\quad C=\begin{bmatrix}0&4&1&9&8&2&7&3&5&6\\3&1&6&8&2&9&4&5&0&7\\6&5&2&4&9&0&3&8&7&1\\1&8&5&3&7&4&9&0&6&2\\9&2&0&5&4&7&8&6&1&3\\8&6&3&7&1&5&0&9&2&4\\4&0&7&2&5&3&6&1&9&8\\2&9&4&1&6&8&5&7&3&0\\7&3&9&6&0&1&2&4&8&5\\5&7&8&0&3&6&1&2&4&9\end{bmatrix}$$

Fig. 4. Pseudotriple of diagonal Latin squares of order 10 from [10]

Fig. 5 presents the corresponding 60 ordered pairs of elements for the pseudotriple from Fig. 4.

$$\begin{bmatrix}00&01&02&-&-&05&06&-&08&-\\10&11&-&13&-&15&16&-&18&-\\-&21&22&-&24&25&-&27&-&29\\-&-&32&33&-&-&36&37&-&39\\-&41&-&-&44&45&46&-&48&49\\50&-&52&53&-&55&-&57&58&59\\60&61&-&-&-&65&66&-&-&69\\-&-&72&73&74&75&-&77&-&79\\-&-&-&83&84&85&-&87&88&-\\90&91&-&93&94&-&96&97&98&99\end{bmatrix}$$

Fig. 5. The set formed by 60 ordered pairs of elements, over which all three pairs of diagonal Latin squares from Fig. 4 are orthogonal

III. Encoding Problems of Search for Systems of Latin Squares to SAT

It is a widely known fact that the system of mutually orthogonal Latin squares as a combinatorial design is equivalent to a number of other combinatorial designs. For example, the pair of MOLS is equivalent to a special set of transversals, to an orthogonal array with some special properties, etc. It means that if we want to construct a system of mutually orthogonal Latin squares, we can do it in a number of various ways using equivalent objects. That is why it is possible to construct vastly different propositional encodings for the same problem. Generally speaking, even when we use one particular representation of a system of orthogonal Latin squares, the predicates involved in the encoding can be transformed to the form "CNF=1" [11] in different ways, thus producing essentially different encodings. Actually, we believe that the impact of the

representation method and techniques used to produce the SAT encodings of the problem on the effectiveness of SAT solvers on corresponding instances is very interesting and we intend to study this question in the nearest future.

In our computational experiments on the search of pairs of orthogonal diagonal Latin squares we used the propositional encoding based on the so called "naive" scheme. It was described, for example, in [12].

Let us briefly describe this encoding. We consider two matrices $A = \|a_{ij}\|$ and $B = \|b_{ij}\|$, $i, j = \overline{1, \dots, n}$. The contents of each matrix cell are encoded via n Boolean variables. It means that one matrix is encoded using n^3 Boolean variables. By $x(i,j,k)$ and $y(i,j,k)$ we denote the variables corresponding to matrices A and B, respectively. Here the variable $x(i,j,k), i,j,k \in \{1, \dots, n\}$ has the value of "Truth" if and only if in the corresponding cell in the i-th row and j-th column of the matrix A there is the number $k-1$. For matrices A and B to represent Latin squares they should satisfy the following constraints on the corresponding variables. Without the loss of generality let us consider these constraints on the example of matrix A.

Each matrix cell contains exactly one number from 0 to $n-1$:

$$\wedge_{i=1}^{n} \wedge_{j=1}^{n} \vee_{k=1}^{n} x(i,j,k);$$
$$\wedge_{i=1}^{n} \wedge_{j=1}^{n} \wedge_{k=1}^{n-1} \wedge_{r=k+1}^{n} \left(\neg x(i,j,k) \vee \neg x(i,j,r)\right).$$

Each number from 0 to $n-1$ appears in each row exactly once:

$$\wedge_{j=1}^{n} \wedge_{k=1}^{n} \vee_{i=1}^{n} x(i,j,k);$$
$$\wedge_{j=1}^{n} \wedge_{k=1}^{n} \wedge_{i=1}^{n-1} \wedge_{r=i+1}^{n} \left(\neg x(i,j,k) \vee \neg x(r,j,k)\right).$$

Each number from 0 to $n-1$ appears in each column exactly once:

$$\wedge_{i=1}^{n} \wedge_{k=1}^{n} \vee_{j=1}^{n} x(i,j,k);$$
$$\wedge_{i=1}^{n} \wedge_{k=1}^{n} \wedge_{j=1}^{n-1} \wedge_{r=j+1}^{n} \left(\neg x(i,j,k) \vee \neg x(i,r,k)\right).$$

In a similar way we write the constraints on the variables forming the matrix B. After this, we need to write the orthogonality condition. For example, we can do it in the following manner:

$$\wedge_{i=1}^{n} \wedge_{j=1}^{n} \wedge_{k=1}^{n} \wedge_{p=1}^{n} \wedge_{q=1}^{n} \wedge_{r=1}^{n} \left(\neg x(i,j,k) \vee \neg y(i,j,k) \vee \neg x(p,q,r) \vee \neg y(p,q,r)\right).$$

Since in the paper we consider not just Latin squares but diagonal Latin squares, we need to augment the described encoding with the constraint, specifying that the primary and secondary diagonals contain all numbers from 0 to $n-1$, where n is the order of the Latin square.

$$\wedge_{i=1}^{n} \vee_{k=1}^{n} x(i,j,k);$$
$$\wedge_{k=1}^{n} \wedge_{i=1}^{n-1} \wedge_{j=\mathrm{i}+1}^{n} (\neg x(i,i,k) \vee \neg x(j,j,k));$$
$$\wedge_{i=1}^{n} \vee_{k=1}^{n} x(i,n-i+1,k);$$
$$\wedge_{k=1}^{n} \wedge_{i=1}^{n-1} \wedge_{j=\mathrm{i}+1}^{n} (\neg x(i,n-i+1,k) \vee \neg x(j,j,k))$$

Also we consider the optimization variant of the problem of search for three MODLS. Since at the present moment it is unknown if there even exist three MODLS we believe that it is natural to weaken this problem and to evaluate the effectiveness of our methods in application to the weakened variant. Among all the constraints that form the corresponding encoding it is most natural to weaken the orthogonality condition. It can be done via different means. For example, one can demand that the orthogonality condition holds only for fixed cells, for fixed ordered pairs, or for the fixed number of cells or for the fixed number of different pairs. In our experiments we weakened the orthogonality condition in the following way. We first fix the parameter $K, K \leq n^2$, called the *characteristics* of the pseudotriple. Then we demand that each two squares in the pseudotriple are orthogonal over the same set of ordered pairs of elements $(a^1, \dots, b^1), \dots, (a^K, \dots, b^K)$.

To consider the corresponding problem in the SAT form, it is necessary to significantly modify the propositional encoding described above. In particular, we have to replace the "old" orthogonality condition with the "new" one. For this purpose we introduce an additional construct: the special matrix $M = \{m_{ij}\}, m_{ij} \in \{0,1\}, i,j \in \{1, \dots, n\}$. We will refer to this matrix as *markings matrix*. We assume that if $m_{ij} = 1$, then for the corresponding pair $(i-1, j-1)$ the orthogonality condition must hold. In the propositional form this constraint is written in the following manner:

$$\wedge_{i=1}^{n} \wedge_{j=1}^{n} (\neg m_{ij} \vee \vee_{p=1}^{n} \vee_{q=1}^{n} (x(p,q,i) \wedge y(p,q,j))).$$

Additionally, if we search for the pseudotriple with the value of characteristics not less than K (it corresponds to the situation when the markings matrix M contains at least K ones) we need to encode the corresponding constraint. For example, it can be done via the following natural manner. First we sort the bits in the Boolean vector $(m_{11}, \dots, m_{1n}, \dots, m_{nn})$ in the descending order, just as we would if it were simply integer numbers. Assume that as the result we obtain the Boolean vector $(\alpha_1, \dots, \alpha_{n^2})$. Then it is clear that the constraint we need would be satisfied if and only if $\alpha_K = 1$. To sort the bits in the Boolean vector it is possible to use various encoding techniques. In our computational experiments we used the CNFs in which it was done using the Batcher sorting networks [13].

IV. Computational Experiments

The problems of search for orthogonal Latin squares using the SAT approach are good candidates for organization of large-scale computational experiments in distributed computing environments. In particular, they suit well for volunteer computing projects [14]. It is explained by the fact that SAT instances on their own allow one to use natural large scale parallelization strategies. In 2011 the authors of the paper in collaboration with colleagues from IITP RAS developed and launched the volunteer computing project SAT@home [8]. This project is designed to solve hard SAT instances from various areas. It is based on the open BOINC platform [15]. As of October, 7, 2015 the project involves 2426 active PCs from participants all over the world. The average performance of SAT@home is about 6 TFLOPs. In subsection 4.1 we describe the experiment performed in SAT@home on the search for new pairs of MODLS of order 10. In subsection 4.2 we use the pairs found on the previous step to search for pseudotriples of diagonal Latin squares of order 10.

4.1. Finding Pairs of Orthogonal Diagonal Latin Squares of Order 10

In 2012 we launched the experiment in SAT@home aimed at finding new pairs of orthogonal diagonal Latin squares of order 10. In this experiment we used the propositional encoding described in the previous section. The client application (the part that works on participants PCs) was based on the CDCL SAT solver MiniSat 2.2 [16] with slight modifications, that made it possible to reduce the amount of RAM consumed.

In the SAT instances to be solved we fixed the first row of the first Latin square to 0 1 2 3 4 5 6 7 8 9 (by assigning

values to corresponding Boolean variables). It is possible because every pair of MODLS can be transformed to such kind by means of simple manipulations that do not violate orthogonality and diagonality conditions. The decomposition of this SAT instance was performed in the following manner. By varying the values in the first 8 cells of the second and the third rows of the first Latin square we produced about 230 billions of possible variants of corresponding assignments, that do not violate any condition. We decided to process in SAT@home only first 20 million subproblems out of 230 billions (i.e. about 0.0087% of the search space). As a result, each subproblem was formed by assigning values to variables corresponding to the first 8 cells of the second and the third rows (with the fixed first row) in the SAT instance considered. The values of remaining 74 cells of the first Latin square and of all cells of the second Latin square were unknown, so the SAT solver had to find it.

To solve each subproblem the SAT solver MINISAT 2.2 had the limit of 2600 restarts that is approximately equal to 5 minutes of work on one core of Intel Core 2 Duo E8400 processor. After reaching the limit the computations were interrupted. In one job batch downloaded by project participant there were 20 of such subproblems. This number was chosen so that one job batch can be processed in about 2 hours on one CPU core (because it suits well for BOINC projects). To process 20 million subproblems (in the form of 1 million job batches) it took SAT@home about 9 months (from September 2012 to May 2013). During this experiment the CluBORun tool [17] was used for increasing performance of SAT@home by engaging idle resources of a computing cluster. The computations for the majority of subproblems were interrupted, but 17 subproblems were solved and resulted in 17 previously unknown pairs of MODLS of order 10 (we compared them with the pairs from [10]). All the pairs found are published on the site of the SAT@home project[1] in the "Found solutions" section. Fig. 6 presents the first pair of MODLS of order 10 found in the SAT@home project.

$$\begin{bmatrix} 0&1&2&3&4&5&6&7&8&9\\ 1&2&0&4&3&7&9&8&5&6\\ 7&3&5&9&0&4&8&6&2&1\\ 3&5&6&8&9&0&4&1&7&2\\ 4&9&7&2&6&8&1&5&0&3\\ 5&8&4&6&7&1&3&2&9&0\\ 8&4&9&1&2&3&7&0&6&5\\ 6&7&3&0&1&2&5&9&4&8\\ 9&0&1&5&8&6&2&4&3&7\\ 2&6&8&7&5&9&0&3&1&4 \end{bmatrix} \quad \begin{bmatrix} 0&1&2&3&4&5&6&7&8&9\\ 7&5&1&9&2&8&0&4&6&3\\ 1&0&3&4&6&7&5&2&9&8\\ 9&8&4&7&5&2&1&0&3&6\\ 6&7&9&0&8&3&2&1&5&4\\ 4&6&5&1&0&9&8&3&2&7\\ 2&3&8&5&1&6&4&9&7&0\\ 5&2&7&8&3&4&9&6&0&1\\ 3&4&6&2&9&0&7&8&1&5\\ 8&9&0&6&7&1&3&5&4&2 \end{bmatrix}$$

Fig. 6. The first pair of MODLS of order 10 found in the SAT@home project.

As we noted in the previous section, one can construct many different propositional encodings for the problem of search for pairs of orthogonal Latin squares. However, in this case the question of comparison of the effectiveness of corresponding encodings becomes highly relevant. The practice showed that the number of variables, clauses and literals usually does not make it possible to adequately evaluate the effectiveness of SAT solvers on corresponding SAT instances. In the nearest future we plan to use the pairs found in the SAT@home project to estimate the effectiveness of different encodings of this particular problem. For each encoding we can construct the set of CNFs (where each CNF corresponds to one known pair of MODLS of order 10) and to make these SAT instances solvable in reasonable time we can weaken them by assigning correct values to Boolean variables corresponding to several rows of the first Latin square of the pair. This series of experiments will make it possible to choose the most effective combination SAT solver + SAT encoding for this particular problem.

4.2. Finding Pseudotriples of Diagonal Latin Squares of Order 10

We considered the following formulation of the problem: to find the pseudotriple of diagonal Latin squares of order 10 with the characteristics value larger than that of the pseudotriple from [10] (see section 2).

On the first stage of the experiment using the encodings described above we constructed the CNFs, in which there was encoded the constraint that the value of characteristics K (see section 3) is greater or equal to the number varied from 63 to 66 with step 1 (i.e. we considered 4 such CNFs). In computational experiments we used the parallel SAT solvers PLINGELING and TREENGELING [18]. Our choice is motivated by the fact that on the SAT competition 2014 these solvers rated in top 3 in parallel categories "Parallel, Application SAT+UNSAT" and "Parallel, Hard-combinatorial SAT+UNSAT". The experiments were carried out within the computing cluster "Academician V.M. Matrosov" of Irkutsk supercomputer center of SB RAS. Each computing node of this cluster has two 16-core AMD Opteron 6276 processors. Thus, each of the SAT solvers mentioned was launched in multithreaded mode on one computing node, i.e. it employed 32 threads. We used time limit 1 day for per instance. Table 1 shows the results obtained using these solvers in application to 4 SAT instances considered with time limit of 1 day per instance.

K	63	64	65	66
PLINGELING	1 h 10 m	1 h 29 m	1 h 21 m	> 1 day
TREENGELING	2h 2 m	4h 8 m	> 1 day	> 1 day

Table 1. The runtime of SAT solvers applied to CNFs encoding the search for pseudotriples with different constraints on the value of characteristics (K).

As a result of the experiments of the first stage we found the pseudotriple with the characteristics value 65 that is better than that of the pseudotriple from [10] (its characteristics value is 60). Note that the runtime of all considered solvers on the CNF encoding the search for a pseudotriple with characteristics value 66 was greater than 1 day (that is why the computations were interrupted and no results were obtained), and by this time each of the solvers consumed all available RAM of the computing node (64 Gb) and started using swap.

On the second stage of our experiments on the search for pseudotriples we used the previously found pairs of MODLS of order 10 (3 pairs from [10] and 17 pairs found in SAT@home). For a fixed value of characteristics K we formed 20 CNFs by assigning values to the Boolean variables corresponding to first two squares (i.e. for each pair of MODLS and for each value of K we constructed one such CNF). Thus each of the CNFs encoded the following problem: for two fixed orthogonal diagonal Latin squares to

find a diagonal Latin square such that in total they form the pseudotriple with characteristics value $\geq K$.

We considered 6 values of parameter K – from 65 to 70. It means that in total we constructed 120 SAT instances (20 for each value of K). Each of two considered solvers was launched on all these CNFs on one computing node of the cluster with time limit of 1 hour per instance. Table 2 shows how many SAT instances out of the family of 20 could the SAT solver solve within the time limit. As a result we managed to find the pseudotriple with the characteristics value 70.

K	65	66	67	68	69	70
PLINGELING	15	11	11	1	3	0
TREENGELING	20	19	15	3	11	1

Table 2. The number of SAT instances, encoding the search for pseudotriples of diagonal Latin squares with two known squares, that the solver managed to solve within the time limit of one hour (out of 20 SAT instances)

The experiments of this stage required substantial computational resources since the amount of SAT instances was quite large. To search for pseudotriples with characteristics greater than 70 we chose the solver that performed best on the second stage. As it is easy to see from the Table 2 it was TREENGELING. On the third stage we launched this SAT solver on 80 SAT instances encoding the search for pseudotriples with two known squares and K varying from 71 to 74. The time limit was increased to 10 hours. As a result we found pseudotriple with characteristics 73. On all 20 SAT instances with $K = 74$ the solution was not found before the time limit. Fig. 7 presents the record pseudotriple with characteristics value 73.

$$A = \begin{bmatrix} 0&1&2&3&4&5&6&7&8&9 \\ 1&2&0&4&3&8&7&9&5&6 \\ 5&6&9&0&7&3&4&8&1&2 \\ 9&8&7&5&6&4&0&1&2&3 \\ 3&7&5&9&8&1&2&6&4&0 \\ 7&5&1&8&2&6&9&3&0&4 \\ 2&4&8&7&1&9&3&0&6&5 \\ 8&9&6&1&0&2&5&4&3&7 \\ 6&3&4&2&9&0&1&5&7&8 \\ 4&0&3&6&5&7&8&2&9&1 \end{bmatrix} B = \begin{bmatrix} 0&1&2&3&4&5&6&7&8&9 \\ 7&4&8&0&5&9&2&3&6&1 \\ 4&2&5&9&3&7&8&1&0&6 \\ 6&0&4&7&9&1&3&8&5&2 \\ 9&6&1&8&2&4&0&5&3&7 \\ 1&3&9&6&7&8&4&0&2&5 \\ 8&9&3&5&6&2&1&4&7&0 \\ 5&7&0&2&1&3&9&6&4&8 \\ 3&8&7&1&0&6&5&2&9&4 \\ 2&5&6&4&8&0&7&9&1&3 \end{bmatrix} C = \begin{bmatrix} 1&3&4&6&0&8&5&7&9&2 \\ 8&6&2&4&7&1&0&5&3&9 \\ 6&8&7&9&3&5&1&4&2&0 \\ 5&9&1&0&4&6&8&2&7&3 \\ 4&2&5&7&8&9&3&0&1&6 \\ 9&7&0&2&5&3&4&8&6&1 \\ 0&4&9&1&6&7&2&3&8&5 \\ 3&1&6&5&2&0&7&9&4&8 \\ 2&0&3&8&9&4&6&1&5&7 \\ 7&5&8&3&1&2&9&6&0&4 \end{bmatrix}$$

Fig. 7. New pseudotriple of diagonal Latin squares of order 10 with characteristics value 73

Fig. 8 shows the corresponding 73 ordered pairs of elements over which the orthogonality condition holds for all pairs of Latin squares from the triple.

$$\begin{bmatrix} - & 01 & 02 & 03 & 04 & 05 & 06 & - & 08 & 09 \\ 10 & - & 12 & 13 & 14 & 15 & 16 & - & 18 & 19 \\ 20 & - & - & 23 & 24 & 25 & 26 & 27 & 28 & - \\ 30 & - & 32 & 33 & 34 & 35 & 36 & 37 & 38 & - \\ 40 & 41 & - & 43 & 44 & - & 46 & 47 & - & 49 \\ 50 & 51 & - & 53 & - & 55 & 56 & 57 & 58 & - \\ 60 & - & 62 & 63 & 64 & 65 & 66 & - & 68 & 69 \\ 70 & 71 & - & 73 & - & 75 & - & 77 & 78 & 79 \\ - & 81 & 82 & 83 & - & - & - & 87 & 88 & 89 \\ 90 & 91 & 92 & - & 94 & 95 & - & 97 & - & 99 \end{bmatrix}$$

Fig. 8. The set formed by 73 ordered pairs of elements, over which all three pairs of squares from Fig. 7 are orthogonal

Note that this pseudotriple is based on one of the 17 pairs of MODLS of order 10 found in the SAT@home project (in the figure the first two squares correspond to the pair found in SAT@home).

V. RELATED WORK

The predecessor of the SAT@home was the BNB-Grid system [19], [20]. Apparently, [21] became the first paper about the use of a desktop grid based on the BOINC platform for solving SAT. It did not evolve into a publicly available volunteer computing project (like SAT@home did). The volunteer computing project with the most similar to SAT@home problem area is Sudoku@vtaiwan [22]. It was used to confirm the solution of the problem regarding the minimal number of clues in Sudoku, previously solved on a computing cluster [4]. In [6] there was described the unsuccessful attempt to solve the problem of search for three MOLS of order 10 using the PSATO SAT solver in a grid system.

VI. CONCLUSION

In the paper we describe the results obtained by applying the resources of the volunteer computing project SAT@home to searching for systems of diagonal Latin squares of order 10. We reduce the original combinatorial problem to the Boolean satisfiability problem, then decompose it and launch solving of corresponding subproblems in SAT@home. Using this approach we found 17 new pairs of orthogonal diagonal Latin squares of order 10. Based on these pairs, we have found new systems of three partly orthogonal diagonal Latin squares of order 10. In the future we plan to develop new SAT encodings for the considered combinatorial problems and also to find new orthogonal (or partly orthogonal) systems of Latin squares.

ACKNOWLEDGMENT

We thank Alexander Semenov for valuable comments, Mikhail Posypkin and Nickolay Khrapov for their help in maintaining the SAT@home project, and all the SAT@home volunteers for their participation.

REFERENCES

[1] C. J. Colbourn and J. H. Dinitz, *Handbook of Combinatorial Designs*, 2nd ed. (Discrete Mathematics and Its Applications). Chapman & Hall/CRC. 2006.

[2] C. W. H. Lam, L. Thiel, and S. Swiercz, "The non-existence of finite projective planes of order 10," *Canad. J. Math.*, vol. 41, pp. 1117-1123, 1989.

[3] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Mathematical Library, North Holland Publishing Co, 1988.

[4] G. McGuire, B. Tugemann and G. Civario, "There is no 16-clue Sudoku: solving the Sudoku minimum number of clues problem via hitting set enumeration," *Experimental Mathematics*, vol. 23, no. 2, pp. 190–217, 2014.

[5] B. D. Mckay, A. Meynert and W. Myrvold, "Small Latin squares, quasigroups and loops," *J. Combin. Designs*, vol. 15(2), pp. 98–119, 2007.

[6] H. Zhang, "Combinatorial Designs by SAT Solvers,". in: *Handbook of Satisfiability*, *Frontiers in Artificial Intelligence and Applications*, vol. 185. IOS Press, 2009.

[7] *Handbook of Satisfiability*, *Frontiers in Artificial Intelligence and Applications*, vol. 185. IOS Press, 2009.

[8] M. A. Posypkin, A. A. Semenov and O. S. Zaikin, "Using BOINC desktop grid to solve large scale SAT problems," *Computer Science Journal*, vol. 13, no. 1, pp. 25–34, 2012.

[9] J. Egan and I. M. Wanless, "Enumeration of MOLS of small order," CoRR abs/1406.3681v2.

[10] Brown et al, "Completion of the spectrum of orthogonal diagonal Latin squares," *Lect. Notes Pure Appl. Math.*, vol. 139, pp. 43–49 (1993)

[11] S. D. Prestwich, “CNF encodings,” in: *Handbook of Satisfiability*, *Frontiers in Artificial Intelligence and Applications*, vol. 185. IOS Press, 2009, pp. 75–97.
[12] I. Lynce and J. Ouaknine, “Sudoku as a SAT problem,” in: *Proc. Ninth International Symposium on Artificial Intelligence and Mathematics*, Fort Lauderdale, FL., 2006.
[13] K. E. Batcher, “Sorting networks and their applications,” in *Proc. of Spring Joint Computer Conference*, New York, USA, 1968, pp. 307–314.
[14] M. Nouman Durrani and J. A. Shamsi, “Review: Volunteer computing: Requirements, challenges, and solutions,” *J. Netw. Comput. Appl.*, vol. 39, pp. 369–380, 2014.
[15] D. P. Anderson and G. Fedak, “The computational and storage potential of volunteer computing,” in *6th IEEE International Symposium on Cluster Computing and the Grid*, Singapore, 2006, pp. 73–80.
[16] N. E´en and N. S¨orensson, “An extensible SAT-solver,” in *6th International Conference on Theory and Applications of Satisfiability Testing*, Santa Margherita Ligure, Italy, 2003, pp. 502–518.
[17] A. P. Afanasiev, I. V. Bychkov, M. O. Manzyuk, M. A. Posypkin, A. A. Semenov and O. S. Zaikin, “Technology for Integrating Idle Computing Cluster Resources into Volunteer Computing Projects,” in *Proc. of The 5th International Workshop on Computer Science and Engineering*, Moscow, Russia, 2015, pp. 109-114.
[18] A. Biere, “Lingeling essentials, A tutorial on design and implementation aspects of the the SAT solver lingeling,” in *Proc. Fifth Pragmatics of SAT workshop*, Vienna, Austria, 2014, p. 88.
[19] Y. Evtushenko, M. Posypkin and I. Sigal. “A framework for parallel large-scale global optimization,”. *Computer Science - Research and Development*, vol. 23(3-4), pp. 211-215, 2009.
[20] A. Semenov, O. Zaikin, D. Bespalov and M. Posypkin, “Parallel Logical Cryptanalysis of the Generator A5/1 in BNB-Grid System,” in *Parallel Computational Technologies*, Kazan, Russia, 2011, pp. 473–483.
[21] M. Black and G. Bard, “SAT Over BOINC: An Application-Independent Volunteer Grid Project,” in *12th IEEE/ACM International Conference on Grid Computing*, Lyon, France, 2011, pp. 226–227.
[22] H.-H. Lin and I-C. Wu, “An Efficient Approach to Solving the Minimum Sudoku Problem,” in. *Proc. International Conference on Technologies and Applications of Artificial Intelligence*, Hsinchu, Taiwan. 2010, pp. 456–461.