SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2013, Vol. 56 032112:1–032112:9
doi: 10.1007/s11432-011-4343-3

# Finding orthogonal latin squares using finite model searching tools

MA FeiFei & ZHANG Jian*

*State key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing* 100190*, China*

Received April 22, 2010; accepted August 23, 2010; published online September 7, 2011

**Abstract** An important class of problems in combinatorics is to find orthogonal latin squares with certain properties. Computer search is a promising approach for solving such problems. But generally its worst-case complexity is high. This paper describes how to use a general-purpose model searching program to find orthogonal latin squares. New techniques for problem representation and symmetry breaking are proposed to increase search efficiency.

**Keywords** latin squares, finite models, symmetry

**Citation** Ma F F, Zhang J. Finding orthogonal latin squares using finite model searching tools. Sci China Inf Sci, 2013, 56: 032112(9), doi: 10.1007/s11432-011-4343-3

## 1 Introduction

The latin square is a fascinating combinatorial structure, attracting much attention from mathematicians and computer scientists [1]. It was introduced by Leonhard Euler in 1783 as a new kind of magic squares, and plays an important role in various fields, such as in combinatorics, statistics and informatics.

Latin squares have been studied by many mathematicians for more than two hundred years. They have developed a rich set of methods for constructing latin squares with certain properties (or proving their non-existence). But there are still a lot of open problems, among which the existence of orthogonal latin squares is a very challenging one.

Euler was the first to state the problem. He asked if it is possible to arrange 36 officers drawn from 6 ranks and 6 regiments in a square so that in each row and column there are 6 officers of different ranks and different regiments. It is known as the famous 36 officer problem, and the square, if exists, is equivalent to two orthogonal latin squares of order 6. Euler went on to conjecture that such a square does not exist for $n = 6$, nor does one exist whenever $n \equiv 2 \pmod 4$. In 1959, Bose, Shrikhande and Parker disproved Euler's conjecture [2], which was quite astonishing at that time. Nevertheless, the orthogonal latin square problem is far from being completely resolved. It remains a great challenge to find out the spectrum of orthogonal latin squares of certain sizes, i.e., the number of latin squares that are mutually orthogonal. This problem is not only of great importance to combinatorics, but also closely related to other fields such as quantum informatics [3].

*Corresponding author (email: zj@ios.ac.cn)

 

As computers become more and more powerful, they have also been used for finding latin squares (esp. when the size of the square is small). See for example [4, 5]. To solve a specific problem in combinatorics, one can either design a special-purpose algorithm and tool, or use a general-purpose search program. In this paper, we shall focus on the latter approach. In particular, we are interested in satisfiability checkers for the classical logics (or finite model searchers for the first-order predicate logic, e.g. SEM [6]). We shall discuss some ideas that will be helpful to find orthogonal latin squares using these tools.

## 2 Basic concepts

In this section, we recall some basic definitions and notations. They are often used in the literature of combinatorial designs. See, for example, [7].

**Definition 1.** A latin square of order $n$ is an $n \times n$ array in which each cell is an element of an $n$-set $\mathcal{D}$, such that each element occurs exactly once in each row and column.

In this paper, we assume that $\mathcal{D} = \{0, 1, \ldots, n-1\}$. We also introduce two other $n$-sets $\mathcal{R}$ and $\mathcal{C}$, and let $\mathcal{R}$ be the set of row indices and $\mathcal{C}$ the set of column indices. For a latin square $L$, we denote by $L(i, j)$ the cell value (or the element) on the $i$th row and the $j$th column.

**Definition 2.** A quasigroup is an ordered pair $(Q, \otimes)$, where $Q$ is a set and $\otimes$ is a binary operation on $Q$ such that the equations $a \otimes x = b$ and $y \otimes a = b$ are uniquely solvable for every pair of elements $a$, $b$ in $Q$.

Here the operator $\otimes$ may have many choices, for instance, the subtraction over integers, or the division over nonzero reals. In this paper, we only discuss quasigroups on finite set. The multiplication table of such a quasigroup defines a latin square.

**Definition 3.** A pair of Latin squares of order $n$ is said to be orthogonal if the $n^2$ pairs of elements formed by juxtaposing the two squares are all distinct. More formally, two latin squares $L_1$ and $L_2$ are said to be orthogonal if for any $x, y \in \mathcal{R}$, and $z, w \in \mathcal{C}$,

$$(L_1(x, z) = L_1(y, w) \wedge L_2(x, z) = L_2(y, w)) \rightarrow (x = y \wedge z = w).$$

Here $\wedge$ denotes conjunction (AND), $\rightarrow$ denotes implication. If two latin squares are orthogonal, one is called the orthogonal mate of the other.

**Example 1.** $\mathrm{LS}_1$ and $\mathrm{LS}_2$ are two orthogonal latin squares. When juxtaposed, we can see that all ordered pairs of cells are distinct.

| | | |
|---|---|---|
| 1 2 3 0 | 3 2 1 0 | 13 22 31 00 |
| 3 0 1 2 | 2 3 0 1 | 32 03 10 21 |
| 2 1 0 3 | 0 1 2 3 | 20 11 02 33 |
| 0 3 2 1 | 1 0 3 2 | 01 30 23 12 |
| $\mathrm{LS}_1$ | $\mathrm{LS}_2$ | $\mathrm{LS}_1$ and $\mathrm{LS}_2$ juxtaposed |

**Definition 4.** Two quasigroups $(Q, \otimes)$ and $(Q, \odot)$ defined on the same set $Q$ are orthogonal if the equations $x \otimes y = z \otimes w$ and $x \odot y = z \odot w$ together imply $x = z$ and $y = w$.

**Example 2.** Consider two quasigroups $(Q, \otimes)$ and $(Q, \odot)$, where $Q$ is a finite set of integers {0,1,2}, $\otimes$ is defined as $x \otimes y = (x - y) \pmod 3$, and $\odot$ is defined as $x \odot y = (2x - y) \pmod 3$. Their multiplication tables are as follows:

| $\otimes$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 2 | 1 |
| 1 | 1 | 0 | 2 |
| 2 | 2 | 1 | 0 |

| $\odot$ | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 0 | 2 | 1 |
| 1 | 2 | 1 | 0 |
| 2 | 1 | 0 | 2 |

The two quasigroups are orthogonal, since $(x-y) \pmod 3 = (z-w) \pmod 3$ and $(2x-y) \pmod 3 = (2z-w) \pmod 3$ imply that $x \pmod 3 = z \pmod 3$, and further we have $x = z$, $y = w$.

When two quasigroups are orthogonal, their corresponding latin squares are also orthogonal in the usual sense [7]. In the above example, the orthogonality of the latin squares can be easily verified.

**Definition 5.** A set of latin squares is mutually orthogonal, if any two latin squares in the set are orthogonal. In this case, the set is called a set of MOLS.

Given positive integers $n$ and $k$ ($n > 1$, $k > 1$), it is interesting to find $k$ MOLS of size $n$. Such a problem is denoted $k$-MOLS($n$). For $n = 10$ and $k = 3$, the case is still open. In other words, we do not know whether there exist 3 MOLS of order 10. In the sequel, we will mainly use 2-MOLS($n$) to compare different techniques.

**Definition 6.** The transpose of a latin square $L$ is the matrix created by reflecting $L$ by its main diagonal (which starts from the top left).

Obviously the transpose of a latin square is also a latin square.

**Definition 7.** A self-orthogonal latin square (SOLS) is a latin square that is orthogonal to its transpose.

A SOLS of size $n$ is often denoted by SOLS($n$). For example, the following square is a SOLS(4):

$$\begin{array}{cccc} 0 & 2 & 3 & 1 \\ 3 & 1 & 0 & 2 \\ 1 & 3 & 2 & 0 \\ 2 & 0 & 1 & 3 \end{array}$$

**Definition 8.** A SOLSSOM($n$) denotes a self-orthogonal latin square of size $n$ with a symmetric orthogonal mate, which is a pair $(S, M)$ such that $S$ is a SOLS($n$) and $M$ is a symmetric latin square of order $n$ that is orthogonal to both $S$ and the transpose of $S$.

**Definition 9.** A permutation is a bijective mapping from a finite set to itself.

There are two main notations for a permutation. In relation notation, one can just arrange the original ordering of the elements being permuted on a row, and the new ordering on another row, for example,

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 3 & 4 & 2 \end{pmatrix}$$

stands for the permutation $s$ of the set $\{0, 1, 2, 3, 4\}$ defined by $s(0) = 1, s(1) = 0, s(2) = 3, s(3) = 4, s(4) = 2$. Alternatively, a permutation can be denoted by its decomposition in a product of disjoint cycles, and each cycle $(x_1\ x_2 \cdots x_k)$ stands for the permutation that maps $x_i$ to $x_{i+1}$ ($i = 1, \ldots, k-1$) and $x_k$ to $x_1$. For example, the permutation $s$ can also be denoted by (0 1)(2 3 4). The order of the cycles in the product does not matter. In this paper, the cycles are ordered non-descendingly by their lengths.

## 3 Modeling

We use first order logic as the modeling language to formulate the MOLS problem and encode the constraints as a set of first order formulas. In this paper, we assume that all the free variables in a logical formula are universally quantified, unless specified otherwise.

Definition 3 in the previous section indicates a straightforward approach to specifying the $k$-MOLS($n$) problem. For the $i$th ($1 \leqslant i \leqslant k$) latin square $\mathrm{LS}_i$, we introduce a function $f_i : \mathcal{R} \times \mathcal{C} \mapsto \mathcal{D}_i$, where $D_i$ is the element domain of $\mathrm{LS}_i$. The function $f_i$ should satisfy:

$$f_i(x_1, y) \neq f_i(x_2, y) \ \vee \ x_1 = x_2,$$

$$f_i(x, y_1) \neq f_i(x, y_2) \ \vee \ y_1 = y_2.$$

In other words, its multiplication table defines a latin square. The constraints of orthogonality between any two latin squares, $\mathrm{LS}_i$ and $\mathrm{LS}_j$ for example, can be naturally represented by the following clauses:

$$f_i(x_1, y_1) \neq f_i(x_2, y_2) \ \vee \ f_j(x_1, y_1) \neq f_j(x_2, y_2) \ \vee \ x_1 = x_2,$$

$$f_i(x_1, y_1) \neq f_i(x_2, y_2) \ \vee \ f_j(x_1, y_1) \neq f_j(x_2, y_2) \ \vee \ y_1 = y_2.$$

Here $\vee$ denotes the logic operator or, also known as disjunction, which results in true whenever one or more of its operands are true.

In this paper, however, we shall introduce another formulation which seems more effective. The new formulation makes use of the transversal concept and reveals the internal relationship between orthogonal latin squares in a more explicit way.

**Definition 10.** A transversal in a latin square of order $n$ is a set of $n$ cells, one from each row and column, containing each of the $n$ symbols exactly once.

**Theorem 1.** A latin square of order $n$ has an orthogonal mate iff it contains $n$ disjoint transversals [7].

Consider the latin square of order 4:

0 1 2 3
1 2 3 0
2 3 0 1
3 0 1 2

It does not even have a transversal, hence it does not have an orthogonal mate.

The positions of $n$-cells of a transversal in a latin square $L$ can be recorded as a vector, where the $i$th element is the row index of the cell that appears in the $i$th column, e.g., $\boldsymbol{t} = \langle r_0, r_1, \ldots, r_{n-1} \rangle$ which means $\{L(r_i, i) | 0 \leqslant i \leqslant n-1\}$.

**Definition 11.** If a latin square $L$ contains $n$ disjoint transversals, we construct a matrix from the vectors of the transversals (each row of the matrix corresponds to a vector). We call it the transversal matrix of $L$, denoted by $\boldsymbol{T}^{L1)}$.

Without loss of generality, we fix the $j$th row of $\boldsymbol{T}^L$ to be the transversal that contains cell $L(j, 0)$. For example, in Figure 1, $\boldsymbol{T}^L$ is a transversal matrix of $L$. Row 0 in $\boldsymbol{T}^L$ is $\langle 0\ 2\ 4\ 1\ 3 \rangle$, and it represents the transversal $\{L(0,0), L(2,1), L(4,2),\ L(1,3), L(3,4)\}$; Row 1 represents the transversal $\{L(1,0), L(3,1), L(0,2), L(2,3), L(4,4)\}$, and so on.

It is easy to prove that a transversal matrix is also a latin square. Suppose in some column $j$ of $\boldsymbol{T}^L$ there are two cells taking the same value, i.e., $T^L(i_1, j) = T^L(i_2, j) = r$ $(i_1 \neq i_2)$, then the two transversals corresponding to row $i_1$ and $i_2$ intersect at the cell $L(r, j)$, contradicting the definition of transversal matrix. It is also obvious that each element occurs exactly once within the same row of $\boldsymbol{T}^L$.

For a latin square $L$, an orthogonal mate can be obtained from its transversal matrix $\boldsymbol{T}^L$ by assigning one symbol to the cells within the same transversal. Let us still consider the example in Figure 1. We can build an orthogonal mate of $L$, denoted by $L'$, as follows. We set the cells indexed by row $i$ of $\boldsymbol{T}^L$ to element $'i'$. More specifically, the cells indexed by row 0 are $\{L'(0,0), L'(2,1), L'(4,2), L'(1,3), L'(3,4)\}$ and they are all assigned $'0'$; the cells $\{L'(1,0), L'(3,1), L'(0,2), L'(2,3), L'(4,4)\}$ are assigned $'1'$, and so on and so forth. Finally we get $L'$ which is orthogonal to $L$, as illustrated in Figure 1.

**Proposition 1.** A $k$-MOLS($n$) exists iff there exists a latin square of order $n$, LS, which has $k-1$ transversal matrices, and for any two transversal matrices $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$ $(i \neq j)$, it holds that

$$(T_i(t_1, y_1) = T_j(t_2, y_1) \wedge T_i(t_1, y_2) = T_j(t_2, y_2)) \rightarrow y_1 = y_2. \tag{1}$$

*Proof.* In the only if direction, we assume the $k$ latin squares are $\mathrm{LS}_1$, $\mathrm{LS}_2$, …,$\mathrm{LS}_k$. Firstly, from each $\mathrm{LS}_i$ $(1 \leqslant i \leqslant k-1)$ we construct a new $n \times n$ matrix $T_i$. For all $x \in \mathcal{R}$ and $y \in \mathcal{C}$, assign cell $\boldsymbol{T}_i(\mathrm{LS}_i(x, y), y)$ with $x$.

1) The superscript can be omitted for simplicity, if no ambiguity arises.

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | | 0 | 2 | 4 | 1 | 3 | | 0 | 3 | 1 | 4 | 2 | | | |
| 1 | 2 | 3 | 4 | 0 | | 1 | 3 | 0 | 2 | 4 | | 1 | 4 | 2 | 0 | 3 | | | |
| 2 | 3 | 4 | 0 | 1 | | 2 | 4 | 1 | 3 | 0 | | 2 | 0 | 3 | 1 | 4 | | | |
| 3 | 4 | 0 | 1 | 2 | | 3 | 0 | 2 | 4 | 1 | | 3 | 1 | 4 | 2 | 0 | | | |
| 4 | 0 | 1 | 2 | 3 | | 4 | 1 | 3 | 0 | 2 | | 4 | 2 | 0 | 3 | 1 | | | |
| | | $L$ | | | | | | $T^L$ | | | | | | $L'$ | | | | | |

**Figure 1** Transversal matrix.

Obviously $\boldsymbol{T}_i$ is a latin square. Moreover, when constructing $\boldsymbol{T}_i$, if we substitute $\boldsymbol{T}_i(t, y)$ for $x$, we get $\boldsymbol{T}_i(\mathrm{LS}_i(T_i(t, y), y), y) = T_i(t, y)$, and furthermore $\mathrm{LS}_i(\boldsymbol{T}_i(t, y), y) = t$. The equation $\mathrm{LS}_i(\boldsymbol{T}_i(t, y), y) = t$ is an essential property that would be used in the following proof.

We now show that $\boldsymbol{T}_i$ is a transversal matrix of $\mathrm{LS}_k$.

For one thing, each row of $\boldsymbol{T}_i$ represents a transversal of $\mathrm{LS}_k$. Let us examine an arbitrary row vector, say row $t$. The cells it indexes in $\mathrm{LS}_k$ are $\{\mathrm{LS}_k(T_i(t, 0), 0), \mathrm{LS}_k(T_i(t, 1), 1), \ldots, \mathrm{LS}_k(T_i(t, n-1), n-1)\}$. Apparently these cells are distributed in different rows and different columns. Assume that for some $y_1$ and $y_2$ ($y_1 \neq y_2$), we have $\mathrm{LS}_k(T_i(t, y_1), y_1) = \mathrm{LS}_k(T_i(t, y_2), y_2)$. From the construction of $\boldsymbol{T}_i$ it can be derived that $\boldsymbol{T}_i(\mathrm{LS}_i(T_i(t, y_1), y_1), y_1) = T_i(t, y_1)$, and furthermore $\mathrm{LS}_i(T_i(t, y_1), y_1) = t$. Similarly we have $\mathrm{LS}_i(T_i(t, y_2), y_2) = t$. Therefore, we get

$$\mathrm{LS}_k(T_i(t, y_1), y_1) = \mathrm{LS}_k(T_i(t, y_2), y_2) \wedge \mathrm{LS}_i(T_i(t, y_1), y_1) = \mathrm{LS}_i(T_i(t, y_2), y_2).$$

This formula violates the premise that $\mathrm{LS}_i$ and $\mathrm{LS}_k$ are orthogonal, thus it is impossible for $\mathrm{LS}_k(T_i(t, y_1), y_1)$ and $\mathrm{LS}_k(T_i(t, y_2), y_2)$ to take the same value. Hence row $t$ of $\boldsymbol{T}_i$ represents a transversal of $\mathrm{LS}_k$.

For the other, since $\boldsymbol{T}_i$ is a latin square, we have $T_i(t_1, y) \neq T_i(t_2, y)$ for any $t_1$, $t_2$ ($t_1 \neq t_2$) and $y$, and further $\mathrm{LS}_k(T_i(t_1, y), y) \neq \mathrm{LS}_k(T_i(t_2, y), y)$, which implies that any two transversals are disjoint. Therefore, $\boldsymbol{T}_i$ is a transversal matrix of $\mathrm{LS}_k$.

We now prove by contradiction that any two of these transversal matrices, say $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$ ($i \neq j$), would satisfy Formula (1). Suppose there are $t_1$, $t_2$ and $y_1$, $y_2$ ($y_1 \neq y_2$) such that $T_i(t_1, y_1) = T_j(t_2, y_1) \wedge T_i(t_1, y_2) = T_j(t_2, y_2)$. Since $\mathrm{LS}_i(T_i(t_1, y_1), y_1) = t_1$, $\mathrm{LS}_i(T_i(t_1, y_2), y_2) = t_1$, $\mathrm{LS}_j(T_j(t_2, y_1), y_1) = t_2$ and $\mathrm{LS}_j(T_j(t_2, y_2), y_2) = t_2$, it holds that

$$\mathrm{LS}_i(T_i(t_1, y_1), y_1) = \mathrm{LS}_i(T_i(t_1, y_2), y_2) \wedge \mathrm{LS}_j(T_j(t_2, y_1), y_1) = \mathrm{LS}_j(T_j(t_2, y_2), y_2).$$

On the previous assumption, this formula violates the premise that $\mathrm{LS}_i$ and $\mathrm{LS}_j$ are orthogonal. Hence $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$ satisfy Formula (1).

In the if direction, assuming the latin square is LS, we construct an $n \times n$ square $\mathrm{LS}_i$ out of each transversal matrix $\boldsymbol{T}_i$. For all $t \in \mathcal{D}_i$ and $y \in \mathcal{C}$, assign cell $\mathrm{LS}_i(T_i(t, y), y)$ with $t$. Since $\boldsymbol{T}_i$ is a latin square, $\mathrm{LS}_i$ is also a latin square.

We now prove by contradiction that $\mathrm{LS}_i$ is orthogonal to LS. Suppose there are some $x_1$, $x_2$ ($x_1 \neq x_2$) and $y_1$, $y_2$ ($y_1 \neq y_2$) such that

$$\mathrm{LS}(x_1, y_1) = \mathrm{LS}(x_2, y_2) \wedge \mathrm{LS}_i(x_1, y_1) = \mathrm{LS}_i(x_2, y_2).$$

Based on the construction of $\mathrm{LS}_i$, we know there is some $t$ such that $T_i(t, y_1) = x_1$ and $T_i(t, y_2) = x_2$. Because $\mathrm{LS}(T_i(t, y_1), y_1) = \mathrm{LS}(x_1, y_1)$ and $\mathrm{LS}(T_i(t, y_2), y_2) = \mathrm{LS}(x_2, y_2)$, $\mathrm{LS}(x_1, y_1)$ and $\mathrm{LS}(x_2, y_2)$ are two cells in the same transversal indexed by row $t$ of $\boldsymbol{T}_i$. However, in the assumption the two cells are of the same value. Because of this contradiction, the original assumption should be rejected.

For any two squares $\mathrm{LS}_i$ and $\mathrm{LS}_j$ ($i \neq j$), the proof of their orthogonality is quite similar to the above one, and is thus omitted.

As a result, instead of searching for $k$ MOLSs of order $n$, we try to find one latin square and $k-1$ of its transversal matrices satisfying Formula (1). We use function $f: \mathcal{R} \times \mathcal{C} \mapsto \mathcal{D}_k$ to represent the latin

square. For the $i$th ($1 \leqslant i \leqslant k-1$) transversal matrix $\boldsymbol{T}_i$, we introduce a function $f_i : \mathcal{D}_i \times \mathcal{C} \mapsto \mathcal{R}$, where $D_i$ is the element domain of $\mathrm{LS}_i$.

The first order formulas for encoding $k$-MOLS($n$) consist of three parts:

1. $f(x_1, y) = f(x_2, y) \ \rightarrow \ x_1 = x_2$,
   $f(x, y_1) = f(x, y_2) \ \rightarrow \ y_1 = y_2$,
   $f_i(t_1, y) = f_i(t_2, y) \ \rightarrow \ t_1 = t_2$,
   $f_i(t, y_1) = f_i(t, y_2) \ \rightarrow \ y_1 = y_2$.

These formulas specify that $f$ and $f_i$ are latin squares. The first formula implies that in an arbitrary column $y$ of $f$, if any two cells have the same value, then they must be the same cell. Similarly, the second formula implies that each value occurs no more than once within any row $x$ of $f$. For the same reason, the last two formulas indicate that $f_i$ is also a latin square.

2. $f(f_i(t, y_1), y_1) = f(f_i(t, y_2), y_2) \ \rightarrow \ y_1 = y_2$.

$f_i$ is the transversal matrix of $f$.

3. $(f_i(t_1, y_1) = f_j(t_2, y_1) \ \wedge \ f_i(t_1, y_2) = f_j(t_2, y_2)) \ \rightarrow \ y_1 = y_2$.

The latin squares $\mathrm{LS}_i$ and $\mathrm{LS}_j$ constructed from $f_i$ and $f_j$ are orthogonal to each other.

## 4 Symmetry breaking

Two sets of MOLSs are isomorphic if one set can be obtained from the other by a combination of row permutations, column permutations, permutations of elements in each latin square. For instance, Figure 2 illustrates two isomorphic 2-MOLS(4) instances.

Generally, in the search for latin squares and other similar structures, it is important to avoid exploring symmetric subspaces in the search space. There are quite a few methods for breaking symmetries in backtracking search.

For MOLS problems, Appa et al. [8, 9] proposed a specific method for symmetry breaking. Essentially it fixes some variables' values (or value domains) before the search begins. Take the 2-MOLS problem for example. The first row of each latin square is fixed in natural order as well as the first column of the first latin square. Moreover, for the first column of the second latin square, the value domains are reduced. The configuration is demonstrated in Figure 2 in [9].

The symmetry breaking method we propose is also to fix variables in advance as in [9]. But instead of reducing value domains in the first column of the second latin square, we restrict the second column of the first latin square to a few options.

The key observation is that any two columns (rows) of a latin square can be viewed as a permutation of the elements. For example, the first two columns of the first latin square in Figure 2 define a permutation:

$$\begin{pmatrix} 1 & 3 & 2 & 0 \\ 2 & 0 & 1 & 3 \end{pmatrix} = (1\ 2)(3\ 0).$$

It is worth noticing that in such a permutation, there is no identity mapping, i.e., no cycle of length 1. Otherwise, there will be an element appearing twice in the same row (column).

**Proposition 2.** Any $k$-MOLS($n$): $\{\mathrm{LS}_i (1 \leqslant i \leqslant k)\}$ can be transformed to the following canonical form:

1. The elements in the first column of $\mathrm{LS}_i$ are naturally ordered.
2. The elements in the first row of $\mathrm{LS}_1$ are naturally ordered.
3. The first two columns of $\mathrm{LS}_1$ define a permutation as:

$$(0\ 1 \cdots k_1)(k_1+1\ k_1+2 \cdots k_2) \cdots (k_i+1\ k_i+2 \cdots n-1),$$

where the cycles are in non-descending order, i.e., $2 \leqslant k_1 + 1 \leqslant k_2 - k_1 \leqslant \cdots \leqslant k_i - k_{i-1} \leqslant n - k_i - 1$.

*Proof.* Suppose the first two columns of $\mathrm{LS}_1$ define a permutation:

$$(x_0 x_1 \cdots x_{k_1})(x_{k_1+1} x_{k_1+2} \cdots x_{k_2}) \cdots (x_{k_i+1} x_{k_i+2} \cdots x_{n-1}).$$

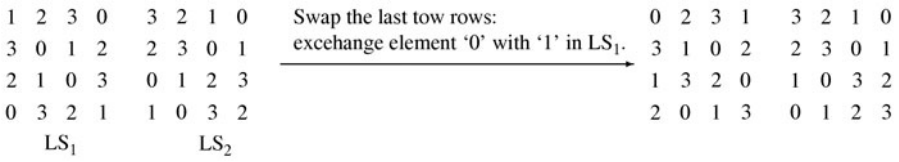


**Figure 2** Two isomorphic 2-MOLS(4) instances.

Firstly, replace each element $x_t$ $(0 \leqslant t \leqslant n-1)$ in $\mathrm{LS}_1$ with $t$. Such a bijective mapping preserves the latin square property of $\mathrm{LS}_1$ while converting the permutation defined by the first two columns of $\mathrm{LS}_1$ to the canonical form.

Secondly, sort the rows in such a way that the first column of $\mathrm{LS}_1$ is $0, 1, \ldots, n-1$. Notice that the rows of each $\mathrm{LS}_i$ must be sorted simultaneously to preserve the orthogonality property. Similarly, the columns are sorted so that the elements in the first row of $\mathrm{LS}_1$ are naturally ordered.

Finally, for each $\mathrm{LS}_i$, apply some element permutation which converts its first column to $\langle 0, 1, \ldots, n-1 \rangle$.

For a concrete example, we now demonstrate how to transform $\mathrm{LS}_1$ and $\mathrm{LS}_2$ in Figure 2 to the canonical form. The first two columns of $\mathrm{LS}_1$ define a permutation (1 2)(3 0). In $\mathrm{LS}_1$, we replace 1 with 0, 2 with 1, 3 with 2 and 0 with 3. $\mathrm{LS}_2$ remains unchanged. We get

$$\begin{array}{cc} 0\,1\,2\,3 & 3\,2\,1\,0 \\ 2\,3\,0\,1 & 2\,3\,0\,1 \\ 1\,0\,3\,2 & 0\,1\,2\,3 \\ 3\,2\,1\,0 & 1\,0\,3\,2 \end{array}$$

Perform a permutation of the set $\mathcal{R}$. More specifically, swap the second row with the third row. Then the 2-MOLS is transformed to

$$\begin{array}{cc} 0\,1\,2\,3 & 3\,2\,1\,0 \\ 1\,0\,3\,2 & 0\,1\,2\,3 \\ 2\,3\,0\,1 & 2\,3\,0\,1 \\ 3\,2\,1\,0 & 1\,0\,3\,2 \end{array}$$

Now $\mathrm{LS}_1$ reaches the canonical form. For $\mathrm{LS}_2$, perform a permutation of its element set $\mathcal{D}_2$: replace 3 with 0, 0 with 1, and 1 with 3. We get the final form, which satisfies all requirements in Proposition 2. It is another instance of 2-MOLS(4).

$$\begin{array}{cc} 0\,1\,2\,3 & 0\,2\,3\,1 \\ 1\,0\,3\,2 & 1\,3\,2\,0 \\ 2\,3\,0\,1 & 2\,0\,1\,3 \\ 3\,2\,1\,0 & 3\,1\,0\,2 \end{array}$$

As for the 2-MOLS(10) problem, the permutation defined by the first two columns of $\mathrm{LS}_1$ can be standardized to one of the 12 patterns listed below:

1. (0 1)(2 3)(4 5)(6 7)(8 9).
2. (0 1)(2 3)(4 5)(6 7 8 9).
3. (0 1)(2 3)(4 5 6)(7 8 9).
4. (0 1)(2 3)(4 5 6 7 8 9).
5. (0 1)(2 3 4)(5 6 7 8 9).
6. (0 1)(2 3 4 5)(6 7 8 9).
7. (0 1)(2 3 4 5 6 7 8 9).
8. (0 1 2)(3 4 5)(6 7 8 9).
9. (0 1 2)(3 4 5 6 7 8 9).
10. (0 1 2 3)(4 5 6 7 8 9).
11. (0 1 2 3 4)(5 6 7 8 9).
12. (0 1 2 3 4 5 6 7 8 9).

**Table 1** CPU time for the 2-MOLS($n$) problem

| $n$ | LS | | QG | |
|---|---|---|---|---|
| | Straightforward | Transversal | Straightforward | Transversal |
| 5 | 0.00 | 0.00 | 0.01 | 0.00 |
| 6 | 92.81 | 1.51 | 2.59 | 0.19 |
| 7 | 13.80 | 0.02 | 17.76 | 0.04 |
| 8 | – | 0.18 | – | 7.69 |
| 9 | – | 55.71 | 131.44 | 0.00 |
| 10 | – | 15.10 | – | 1944.12 |

–: 4 hours' time out.

**Table 2** CPU time for the SOLSSOM problem

| $n$ | Straightforward | Transversal |
|---|---|---|
| 6 | 0.03 | 0.00 |
| 7 | 0.30 | 0.02 |
| 8 | 402.78 | 5.94 |
| 9 | 49094.81 | 1464.62 |

**Table 3** Comparison of two symmetry breaking methods

| 2-MOLS($n$) | SB2 | | | SB1 | |
|---|---|---|---|---|---|
| | # Patterns | # Models | Time | # Models | Time |
| 5 | 2 | 3 | 0.00 | 4 | 0.00 |
| 6 | 4 | 0 | 0.19 | 0 | 1.02 |
| 7 | 4 | 4132 | 980.61 | 6236 | 10922.08 |

It is worth mentioning that the permutations of rows, columns and elements not only preserve the orthogonality of the MOLS but also preserve the number of transversals in each latin square. The cells in a transversal of the original latin square will still form a transversal after a permutation. Since the main diagonal of an idempotent latin square is a transversal, it can be proved that our symmetry breaking technique is also applicable to idempotent latin squares with a slight modification.

## 5 Experimental results

We utilize the first order model generator SEM to search for MOLS. The experiments were performed on an Intel 1.86 GHZ 2 CPU PC with Fedora 7 OS. All timings are given in seconds.

We compare the two different representations on the 2-MOLS problem and the SOLSSOM problem. The results are illustrated in Table 1 and Table 2. Note that there are two distinct ideas concerning how to model a latin square: one is to treat its rows, columns and elements as different domains; the other is to regard the latin square as a quasigroup, that is, all variables share the same domain. Both of the ideas are evaluated in Table 1, with the former labeled "LS" and the latter "QG". However, it seems that there is no clear winner. So in the rest experiments we only list the performance of the latter.

Although the transversal formulation has as many variables as the direct approach, our experiments show that it can significantly reduce the search time.

To compare the pruning power of our symmetry breaking method (denoted by SB2) with the method proposed by Appa et al. (denoted by SB1), we conduct some experiments on the 2-MOLS($n$) problems and compare the numbers of models and running times. Since for order $n \geqslant 8$ the searching process to find all models cannot be completed within reasonable time, we only list the cases where $n \leqslant 7$. It can be seen in Table 3 that our method can eliminate more isomorphic models and significantly reduce the running times.

## 6 Conclusions

In this paper, we have demonstrated how to find MOLS with finite model searching tools. In particular, we have proposed a novel approach to modeling the problem. The basic idea is to search for a latin square with transversal matrices. Experiments show that the new modeling method is more efficient than the ordinary one. Another contribution of the paper is a new symmetry breaking technique for the MOLS problem.

We are also developing other techniques to further accelerate the searching process, including efficient constraint propagation mechanism for the model searching program, more thorough symmetry breaking techniques, as well as parallelization of the program.

**Acknowledgements**

This work was supported by National Natural Science Foundation of China (Grant No. 60673044). We are grateful to Jia Xiangxue for his help in this research. We also thank Zhu Lie and Fang Kaitai for their valuable advices.

## References

1 Denes J, Keedwell A D. Latin Squares and their Applications. New York: Academic Press, 1974. 1–547
2 Bose R C, Shrikhande S S, Parker E T. Further results on the construction of mutually orthogonal latin squares and the falsity of Euler's conjectures. Can J Math, 1960, 12: 189–203
3 Paterek T, Daki'c B, Brukner C. Mutually unbiased bases, orthogonal latin squares, and hidden-variable models. Phys Rev A, 2009, 79: 1–6
4 Lam C W H. The search for a finite projective plane of order 10. Am Math Mon, 1991, 98: 305–318
5 Slaney J, Fujita M, Stickel M. Automated reasoning and exhaustive search: quasigroup existence problems. Comput Math Appl, 1995, 29: 115–132
6 Zhang J, Zhang H. SEM: a system for enumerating models. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence. Burlington: Morgan Kaufmann, 1995. 298–303
7 Colbourn C J, Dinitz J H. Handbook of Combinatorial Designs. 2nd ed. Boca Raton: Chapman & Hall / CRC, 2006. 135–211
8 Appa G, Mourtos I, Magos D. Integrating constraint and integer programming for the OLS problem. LNCS, 2002, 2470: 17–32
9 Appa G, Magos D, Mourtos I. Searching for mutually orthogonal latin squares via integer and constraint programming. Eur J Oper Res, 2006, 173: 519–530