Quasigroups and Related Systems **25** (2017), 73 − 86

Enumeration of exponent three IP loops

Majid Ali Khan, Shahabuddin Muhammad, Nazeeruddin Mohammad, Asif Ali

Abstract. Inverse Property Loops (IP Loops) are important algebraic structures that fall between loops and groups. Enumerating isomorphism classes of higher order IP loops is an arduous task due to enormous number of isomorphism copies. This paper describes a systematic approach to efficiently eliminate isomorphic copies, which reduces the time to enumerate isomorphism classes. Using the proposed approach, we count and enumerate exponent 3 IP loops of order 15. To the best of our knowledge, this count is reported for the first time in the literature. Further, we also computationally verify and enumerate the existing results for exponent 3 IP loops of order up to 13. The results show that even after applying stringent condition of exponent 3, a good number of isomorphism classes exist. However, when associativity property is applied, the total number of isomorphism classes reduces drastically. This provides an insight that instead of exponent 3 property, associativity is mainly responsible for the low population of isomorphism classes in groups.

1. Introduction

A *quasigroup* is a groupoid G with a binary operation $*$ such that $x * a = y$ and $b * x = y$ have unique solutions for each x, $y \in G$. A quasigroup is a *loop* if and only if it contains an *identity element* e such that $x * e = x = e * x$ for each $x \in G$. A loop L is called an *inverse property* (*IP*) *loop* if it has a two sided inverse x^{-1} such that $x^{-1} * (x * y) = y = (y * x) * x^{-1}$ for each x, $y \in L$. A *Steiner loop* is an IP loop of exponent 2 ($x * x = e$ or $x^2 = e$ for all $x \in L$). Also, extensively studied *Moufang loops* are IP loops satisfying $x * (z * (y * z)) = ((x * z) * y) * z$.

IP loops form an important class since they represent a generalization of Steiner loops, Moufang loops, and groups. Further, IP loops represent those groupoids whose power sets are exactly the semi-associative relation algebras [19].

The smallest IP loop which is not a group is of order 7. But the number of IP loops increases quickly with the increase in the order of the loop as there are 10,341 IP loops available for $n = 13$. The IP loops having order greater than 13 are not reported in the literature because of the huge search space. On the other hand, the number of groups does not necessarily increase with the increase in their

2010 Mathematics Subject Classification: 20N05, 05B15
Keywords: 3 IP Loops, Groups, Isomorphism Classes, Associativity, Symmetry Breaking.
This work was partially supported by Prince Mohammad Bin Fahd University (PMU).

order. For example, the number of groups for any given prime order is always one. Enumeration of very highly structured loops like Moufang loops is possible up to comparatively high orders [30], where as less structured loops such as nilpotent loops have not been enumerated so far for higher orders [8].

The IP loops of exponent 3 satisfy the following property: $(x*x)*x = x*(x*x) = e$ for all $x \in L$ (i.e., $x^2 = x^{-1}$). For any order n, the IP loops of exponent 3 exists when either $n \equiv 1 \ (mod\ 6)$ or $n \equiv 3 \ (mod\ 6)$ [31]. Figure 1 shows an example exponent 3 IP loop of order 15.

*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
e=0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	2	0	5	7	4	9	3	11	8	14	13	6	10	12
2	2	0	1	7	5	3	12	4	9	6	13	8	14	11	10
3	3	6	8	4	0	10	2	13	1	12	11	14	7	5	9
4	4	8	6	0	3	13	1	12	2	14	5	10	9	7	11
5	5	11	10	2	1	6	0	9	14	4	12	7	13	8	3
6	6	4	3	14	9	0	5	11	13	7	2	1	10	12	8
7	7	10	12	1	2	14	13	8	0	11	6	3	5	9	4
8	8	3	4	11	14	12	10	0	7	13	1	9	2	6	5
9	9	14	13	6	12	1	11	2	5	10	0	4	8	3	7
10	10	5	7	13	11	8	3	14	12	0	9	6	4	2	1
11	11	13	5	10	8	9	14	1	6	3	7	12	0	4	2
12	12	7	14	9	13	2	8	10	4	5	3	0	11	1	6
13	13	9	11	12	10	7	4	6	3	2	8	5	1	14	0
14	14	12	9	8	6	11	7	5	10	1	4	2	3	0	13

x	x^2	x^{-1}
0	0	0
1	2	2
2	1	1
3	4	4
4	3	3
5	6	6
6	5	5
7	8	8
8	7	7
9	10	10
10	9	9
11	12	12
12	11	11
13	14	14
14	13	13

Figure 1: IP loop of exponent 3 with order 15

The class of elementary abelian p-groups is very small; for example, the total number of abelian 3-groups having order up to 1000 is only seven. It is generally believed that the exponent property ($x^p = e$) is responsible for such a low population of abelian 3-groups. However, we have observed that the number of IP loops of exponent 3 exists in a large quantity; there are 27,765 such IP loops of order less than or equal to 15. This provides us the notion that the exponent property is not keeping the population of groups so small. Rather, we demonstrate that it is the associative property that is reducing the number of groups.

This paper advances counting the history of loops and presents for the first time the count of IP loops of exponent 3 having order 15. The presented results are obtained through enumeration and hence are available for inspection. In this paper, our contributions are as follows:

- We have enumerated, for the first time, the IP loops of exponent 3 having order 15.
- We have compared the associativity and exponent properties in IP loops and concluded that associativity is more stringent than exponent property.

- We have computationally verified and enumerated existing IP loops of exponent 3 having order up to 13.

The rest of the paper is organized as follows. Section 2 describes the history of counting Latin squares and loops. The proposed systematic approach to count isomorphism classes of IP Loops is discussed in Section 3. Results and the related discussions are presented in Section 4.

Key milestones in Latin Square (LS) counting	Historical study details
Reduced LS up to N=5	Euler (1782) [10] Cayley (1890) [7] MacMahon (1915) [18] used a different method to count, but obtained a wrong answer
Reduced LS up to N=6	Frolov (1890) [12] Tarry (1900) [32] Jacob(1930) [15]-incorrectly
Main classes, isotopy classes, and reduced LS up to N=6	Schonhardt (1930) [29]
Isotopy classes up to N=6	Fisher and Yates (1934) [11]
Main classes and isotopy classes for N=7	Norton (1939)[24] -incorrectly Sade (1951) [26] Saxena (1951) [28] using MacMahon's approach
Main classes for N=8	Arlazarov et al (1978) [3]-incorrectly Kolesova et al (1990) [17]
Isotopy classes up to N=8	Brown (1968) [6]-incorrectly Kolesova et al (1990) [17]
Reduced LS for N=8	Wells (1967) [33]
Reduced LS for N=9	Bammel and Rothstein (1975)
Reduced LS for N=10	McKay and Rogoyski (1995) [21]
Reduced LS for N=11	McKay and Wanless (2005) [22]
Main classes and isotopy classes for N=9, 10	McKay, Meynert and Myrvold (2007) [20]
Main classes and isotopy classes for N=11	Hulpke, Kaski and Östergård (2011) [14]

Table 1: History of counting Latin Squares

2. Related work

Earliest history of counting Latin Squares (LS) goes back to at least 1782 as the number of reduced LS of order 5 was known to Euler [10] and Cayley [7]. However, as noted by McKay et al [20], the counting has been constantly troubled by published errors. The history of counting reduced Latin squares and Loops is summarized in Tables 1 and 2. These tables show the main achievements and the related studies.

Key milestones in Isomorphism classes of Loops and Quasigroups counting	Historical study details
Loops up to N = 6	Schonhardt (1930) [29], Albert (1944) [1] and Sade (1970) [27]
Loops up to N = 7	Brant and Mullen (1985) [5]
Loops for N=8	QSCGZ (2001) [25], Guerin (2001) [20]
Loops up to N= 10	McKay, Meynert and Myrvold (2007) [20]
Quasigroups up to N= 6	Bower (2000) [20]
Quasigroups up to N= 10	McKay, Meynert and Myrvold (2007) [20]
Quasigroups and Loops of N= 11	Hulpke, Kaski and Östergård (2011) [14]
Inverse Property Loops up to N=13	Slaney and Ali (2008) [2]

Table 2: History of counting loops and quasigroups

Although researchers had interest in Latin squares, there has been considerable delay in achieving consecutive milestones. This was because of sheer computational complexity of the problem. These historical results were obtained through deduced mathematical formulas [23, 13], applying algorithmic approaches [27, 33, 4] or formulating them as constraint programming problems [2, 9]. In this paper we used constraint programming approach to further explore IP loops. We obtained for the first time the IP loops of exponent 3 of order up to 15. The algorithmic strategies applied to overcome the computational complexity to obtain these results are discussed in the following sections.

3. Enumerating isomorphism classes of IP loops

In order to count the number of IP loops of any order, we model the system as finite domain constraint satisfaction problem (CSPs), where the range of the binary operation $*$ is a CSP variable whose domain consists of elements of the algebra. Then the constraints related to Latin square, loop, and IP loop properties are applied on CSP variables. Constraint solver explores the state space in order to find all possible solutions that satisfy the specified constraints. For higher orders (even for order greater than 10) the state space becomes too large to perform exhaustive search for all IP loops. Therefore, we added more constraints for symmetry breaking which resulted into reduced state space. The constraints used for symmetry breaking along with other constraints are given in Table 3.

The solutions generated by constraint solver have enormous number of isomorphic copies. These redundant isomorphic copies need to be eliminated in order to get the count of isomorphism classes. The following subsection describes the techniques used to eliminate isomorphic copies from these solutions.

No.	Name	Constraint
1	Latin square	$\forall row : \forall i, j \in row, x_i = x_j \Rightarrow i = j$ $\forall col : \forall i, j \in col, y_i = y_j \Rightarrow i = j$
2	Loop	$\forall x : e * x = x = x * e$
3	IP loop	$\forall x, y \in L : x^{-1}*(x*y) = (y*x)*x^{-1} = y$
4	Basic symmetry breaking in IP loop	$\lvert x - x^{-1}\rvert \leq 1$
5	Odd and even symmetry breaking	Odd/Even symmetry breaking constraints of [2]
6	Isomorphism	$*_1$ Isom.$*_2 \Leftrightarrow \forall i, j \in *_1,\ f(i *_1 j) = f(i) *_2 f(j)$
7	Exponent 3	$\forall x : (x * x) * x = e = x * (x * x)$
8	Group	$\forall x, y, z : (x * y) * z = x * (y * z)$

Table 3: Constraints for exponent 3 IP loops and symmetry breaking

3.1. Valid mapping generation

Given two IP Loops $(L_1, *)$ and $(L_2, .)$, finding whether these loops are isomorphic to each other boils down to checking if there exists a bijective function $f : L_1 \rightarrow L_2$ such that for all u and v in L_1: $f(u * v) = f(u).f(v)$. In our case, L_1 $(n \times n)$ is isomorphic to L_2 $(n \times n)$ if $\forall i, j \leq n$, $f(L_1[i][j]) = L_2[f(i)][f(j)]$. Here f is any permutation of $1...n$ elements. Finding isomorphism in this way, by applying the above formula for all permutations of f is extremely time consuming and involves huge number of possibilities for even slightly large n. However, we observed that there are many permutations (mappings) of f which do not satisfy the isomorphic relation $f(m_1[i][j]) = m_2[f(i)][f(j)]$ for all values of $i, j \leq n$ because of constraints shown in Table 3. We consider these mappings as invalid and discard them. We use constraint solver to find all valid mappings which satisfy isomorphic relationship between two IP Loops. Figure 2 represents valid mapping generation process.

The constraint solver models the system by specifying the relevant constraints

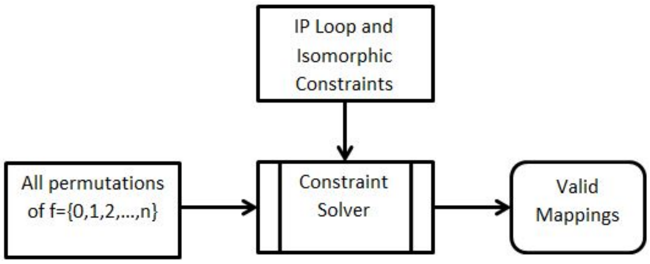

Figure 2: Schematic diagram of valid mappings generator

from Table 3. After the constraints are embedded in the model, the constraint solver searches the state space to find those permutations that satisfy these constraints. All such permutations are called "valid mappings". If the set S represents all the permutations of f and the set S_v represents all the valid maps then $S_v \subseteq S$. The obtained valid mappings are then used to find isomorphism classes.

Figure 3 shows an example of invalid mapping $f(4\ 5)$. This mapping, if applied to a valid IP loop structure (shown on left) will produce an algebraic structure (shown on right) which does not satisfy the basic symmetry breaking constraint (i.e., $|x - x^{-1}| \leq 1$). For example, in the algebraic structure on the right side, for $x = 3$; $x^{-1} = 5$ and thus $|x - x^{-1}| > 1$.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	0	5	7	4	9	3	11	8	14	13	6	10	12
2	0	1	7	5	3	12	4	9	6	13	8	14	11	10
3	6	8	4	0	10	2	13	1	12	11	14	7	5	9
4	8	6	0	3	13	1	12	2	14	5	10	9	7	11
5	11	10	2	1	6	0	9	14	4	12	7	13	8	3
6	4	3	14	9	0	5	11	13	7	2	1	10	12	8
7	10	12	1	2	14	13	8	0	11	6	3	5	9	4
8	3	4	11	14	12	10	0	7	13	1	9	2	6	5
9	14	13	6	12	1	11	2	5	10	0	4	8	3	7
10	5	7	13	11	8	3	14	12	0	9	6	4	2	1
11	13	5	10	8	9	14	1	6	3	7	12	0	4	2
12	7	14	9	13	2	8	10	4	5	3	0	11	1	6
13	9	11	12	10	7	4	6	3	2	8	5	1	14	0
14	12	9	8	6	11	7	5	10	1	4	2	3	0	13

$:f(4\ 5)$
$\approx$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	0	4	5	7	9	3	11	8	14	13	6	10	12
2	0	1	7	3	4	12	5	9	6	13	8	14	11	10
3	6	8	5	10	0	2	13	1	12	11	14	7	4	9
4	11	10	2	6	1	0	9	14	5	12	7	13	8	3
5	8	6	0	13	3	1	12	2	14	4	10	9	7	11
6	5	3	14	0	9	4	11	13	7	2	1	10	12	8
7	10	12	1	14	2	13	8	0	11	6	3	4	9	5
8	3	5	11	12	14	10	0	7	13	1	9	2	6	4
9	14	13	6	1	12	11	2	4	10	0	5	8	3	7
10	4	7	13	8	11	3	14	12	0	9	6	5	2	1
11	13	4	10	9	8	14	1	6	3	7	12	0	5	2
12	7	14	9	2	13	8	10	5	4	3	0	11	1	6
13	9	11	12	7	10	5	6	3	2	8	4	1	14	0
14	12	9	8	11	6	7	4	10	1	5	2	3	0	13

Figure 3: Example of an invalid mapping which produces an algebraic structure (on the right) that does not satisfy the basic symmetry breaking constraint ($|x - x^{-1}| \leq 1$)

Detecting isomorphism classes using valid mappings drastically increases the efficiency because S_v is usually much smaller than S. For example, for IP loop of order 15, the possible number of mappings ($|S|$) is approximately 87 billion but there are only 509,086 valid mappings (i.e., $|S_v|$ is 0.0005% of $|S|$). This results in much faster isomorphic detection.

Table 4 shows the reduced number of valid mappings and their impact on the time taken to identify isomorphism classes for three different problems. In all the three cases, we observed considerable improvement in time when detecting isomorphism. This improvement is even more significant for higher order IP loops. For example, for IP loop of order 11, the time taken to identify isomorphism classes is reduced by a factor of 500 when valid mappings were used.

Algebraic Structure	**Total Solutions**	**Isomorphism Classes**	$\lvert S\rvert$	$\lvert S_v\rvert$	**Time with S (sec)**	**Time with S_v (sec)**	**Time with S_v and Tree (sec)**
Latin Square (Order 5)	161280	1411	120	120	138	138	2
IP Loop (Order 11)	6464	49	3628800	3654	5085	10	5
IP Loop of exponent 3 (Order 13)	22000	64	$\approx 479 \times 10^6$	34804	$\approx$ 864000	571	185

Table 4: Time reductions obtained using valid mappings and tree-based approaches

3.2. Tree representation of isomorphism classes

In order to identify a new isomorphism class, we need to check a newly found solution against all the previously found isomorphism classes using all valid mappings. This results in a large number of computations, and even with the reduced set of mappings the computational time was too high. After careful examination of isomorphism classes we discovered that these classes have similar structure (elements), and with proper organization of isomorphism classes several computations can be eliminated. So we devised a scheme that represents isomorphism classes using a tree-based structure to reduce redundant computations.

The tree structure is built such that each branch of the tree represents one isomorphism class. All new isomorphism classes are added to the existing tree. As long as two isomorphism classes have the same element values, they are represented by a single branch in the tree. If element values differ at any depth in a branch, a new offshoot is created to represent all the subsequent values.

This representation reduces the memory needed to maintain isomorphism classes, especially when the number of isomorphism classes are high. In addition to memory saving, the tree-based approach drastically improves the speed in detecting isomorphism classes in two ways. First, by eliminating redundant computations since one node in the tree represents elements of many isomorphism classes. Second, by discarding all the siblings of a node whenever it satisfies the isomorphism constraint.

The last column in Table 4 (i.e., Time with $|S_v|$ and Tree) shows the results obtained by using tree representation on different problems. As anticipated, the reduction in time depends on the number of isomorphism classes. For example the time taken to detect isomorphism classes with tree representation was reduced by a factor of 2 when the number of isomorphism classes was 45, whereas the time taken was reduced by a factor of 45 when the number of isomorphism classes was 6808.

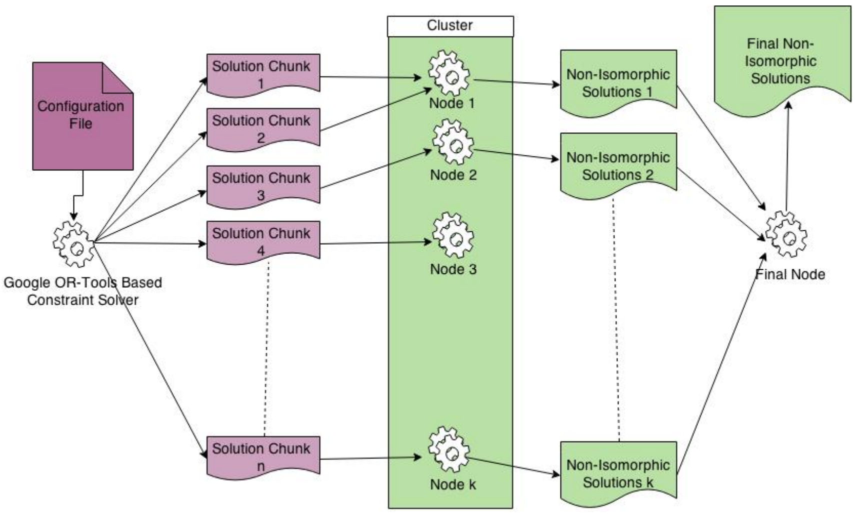

Figure 4: Proposed distributed system to identify isomorphism classes

3.3. Distributed system

With the help of reduced mappings and tree representation, isomorphism classes of IP loops up to order 13 can be enumerated in reasonable time using a single desktop machine. However for higher orders, even after reducing the set of mappings and the number of comparisons, the number of isomorphic copies are still too high to be managed by a standalone system. To cope up with this problem, we developed a distributed system for identifying isomorphism classes as shown in Figure 4. The distributed system takes a single input file containing solutions provided by the constraint solver and breaks it up into several files each containing a manageable subset of the solutions. Each node (i.e., a processor) in the distributed system selects one of the input files for exclusive use and produces the isomorphism classes using valid mappings and tree representation as described in previous sections. The output is written into an intermediate file for further processing. These output files can still contains isomorphic copies as the nodes are unaware of the isomorphism classes found by each other. Therefore, another node exhaustively searches all the intermediate files to produce the final set of isomorphism classes.

Order	**Total solutions**	**Isomorphism classes**		$\lvert S_v\rvert$ ($\lvert S\rvert$)	**Time (sec)**
		non-associative	associative (groups)		
5	0	0	0	0 (24)	< 1
7	2	1	0	48 (720)	< 1
9	10	1	1	276 (40,320)	< 1
11	0	0	0	2402 (3,628,800)	< 1
13	$22,000$	64	0	43804 (≈ 479 million)	210
15	$71,149,968$	$27,698$	0	509086 (≈ 87 billion)	$334,725$ ≈ 93 hours (for total solutions)

Table 5: IP loops of exponent 3

4. Results and discussions

We modeled the system as finite domain CSPs and used a generic constraint solver JaCoP to generate IP loops. We were able to verify the results up to IP loops of order 11 using JaCoP. However, we encountered severe memory and latency issues for higher orders. Therefore, we tried another leading constraint solver Google's or-tools and were able to resolve the memory and latency issues. We modeled all IP loop constraints in or-tools and enumerated IP loops of higher order. The valid mappings and tree representation were used to speed up the process of finding isomorphism classes.

The results for IP loops of exponent 3 are shown in Table 5. We have verified the known results till order 13 and produced new results for order 15. For IP loops of exponent 3 having order 13, or-tools constraint solver produced 22,000 solutions. It took 210 seconds on a general desktop to find all the 64 isomorphism classes.

For IP loops of exponent 3 having order 15, constraint solver produced roughly 71 million solutions. It took about 28 hours to get these results. 27,698 isomorphism classes were found by using the distributed system described in Section 3.3. It was executed on 71 different processors on 20 general desktop computers. It took about 4 days to find the complete set of isomorphism classes.

Generating IP loops of higher orders gave us new perspective about the algebraic structures and their properties. As shown in the Table 5, the number of non-associative isomorphism classes has a reasonable size for higher orders. However, their size plummets to very small number as soon as associativity property is added to the structure. This clearly indicates that it is the associativity property that is seldom present in algebraic structures thus drastically reducing the number of isomorphism classes.

Size of automorphism group	**Number of exponent 3 IP loops**
1	25899
2	1385
3	171
4	140
6	50
8	22
12	10
16	2
21	3
24	13
168	1
192	1
1344	1

Table 6: Size of automorphism group of exponent 3 IP loops of order 15

We have also computed the size of automorphism groups of exponent 3 IP loops of order 15 which is shown in Table 6. The IP loop with largest automorphism group is shown in Figure 5.

Another interesting thing to note is the count of 3×3 Latin subsquares in exponent 3 IP loops due to their role in a conjecture by van Rees [16]. Table 7 shows the count of 3×3 Latin subsquares in exponent 3 IP loops and Figure 6 shows two exponent 3 IP loops of order 15 which have the highest count of 3×3 Latin subsquares. Both these loops have 91 such Latin subsquares.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	0	5	6	4	3	9	10	8	7	13	14	12	11
2	0	1	6	5	3	4	10	9	7	8	14	13	11	12
3	6	5	4	0	1	2	11	12	14	13	8	7	9	10
4	5	6	0	3	2	1	12	11	13	14	7	8	10	9
5	3	4	2	1	6	0	14	13	12	11	9	10	7	8
6	4	3	1	2	0	5	13	14	11	12	10	9	8	7
7	10	9	12	11	13	14	8	0	1	2	3	4	6	5
8	9	10	11	12	14	13	0	7	2	1	4	3	5	6
9	7	8	13	14	11	12	2	1	10	0	6	5	4	3
10	8	7	14	13	12	11	1	2	0	9	5	6	3	4
11	14	13	7	8	10	9	4	3	5	6	12	0	1	2
12	13	14	8	7	9	10	3	4	6	5	0	11	2	1
13	11	12	10	9	8	7	5	6	3	4	2	1	14	0
14	12	11	9	10	7	8	6	5	4	3	1	2	0	13

Figure 5: IP loop of exponent 3 with order 15 having the largest automorphism group (size=1344)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	0	5	7	9	11	12	10	3	13	14	4	8	6
2	0	1	9	12	3	14	4	13	5	8	6	7	10	11
3	11	8	4	0	14	2	9	6	13	1	10	5	7	12
4	10	6	0	3	12	8	13	2	7	11	1	14	9	5
5	13	12	7	1	6	0	11	14	8	2	3	10	4	9
6	4	10	11	13	0	5	3	9	14	12	7	2	1	8
7	14	9	1	5	10	13	8	0	11	4	2	6	12	3
8	3	11	14	10	4	12	0	7	2	5	9	13	6	1
9	7	14	12	2	11	1	6	3	10	0	13	8	5	4
10	6	4	8	14	13	7	1	12	0	9	5	3	11	2
11	8	3	13	6	1	9	14	5	4	7	12	0	2	10
12	5	13	2	9	8	4	10	1	6	14	0	11	3	7
13	12	5	6	11	7	10	2	4	1	3	8	9	14	0
14	9	7	10	8	2	3	5	11	12	6	4	1	0	13

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	0	5	7	9	11	12	10	4	13	14	3	6	8
2	0	1	12	9	3	13	4	14	5	8	6	7	10	11
3	10	8	4	0	13	2	5	11	12	6	1	14	9	7
4	11	6	0	3	7	10	14	2	13	1	8	9	5	12
5	14	12	9	1	6	0	11	3	8	2	13	10	7	4
6	4	10	8	14	0	5	13	9	3	12	7	2	11	1
7	13	9	1	12	10	4	8	0	11	14	2	6	3	5
8	3	11	13	6	14	12	0	7	2	5	9	4	1	10
9	7	14	2	5	11	1	6	13	10	0	4	8	12	3
10	6	3	14	11	4	7	1	12	0	9	5	13	8	2
11	8	4	10	13	1	9	3	5	14	7	12	0	2	6
12	5	13	7	2	8	14	10	1	6	3	0	11	4	9
13	12	7	11	8	2	3	9	6	1	4	10	5	14	0
14	9	5	6	10	12	8	2	4	7	11	3	1	0	13

Figure 6: IP loops of exponent 3 with order 15 having the highest count of 3×3 Latin subsquares

Count of 3×3 Latin subsquares	**Number of exponent 3 IP loops**	**Count of 3×3 Latin subsquares**	**Number of exponent 3 IP loops**
7	992	34	44
9	856	35	34
10	2083	36	25
11	457	37	59
12	1996	38	20
13	2676	39	10
14	1046	40	19
15	2430	41	14
16	2440	42	4
17	1279	43	32
18	2022	45	5
19	1977	46	5
20	988	47	2
21	1397	48	3
22	1090	49	14
23	619	50	2
24	705	51	4
25	626	52	3
26	332	53	1
27	422	55	13
28	293	58	1
29	175	61	1
30	159	67	1
31	177	73	1
32	62	91	2
33	80		

Table 7: Number of exponent 3 IP loops of order 15 grouped by count of 3×3 Latin subsquares

References

[1] **A.A. Albert**, *Quasigroups. II*, Trans. American Mathematical Society **55** (1944), 401 – 409.

[2] **A. Ali and J. Slayney**, *Counting loops with the inverse property*, Quasigroups and Related Systems **16** (2008), 13 – 16.

[3] **V.L. Arlazarov, A.M. Baraev, Y.Y. Gol'fand and I.A. Faradzev**, *Construction with the aid of a computer of all latin squares of order* 8, Algorithmic Investigations in Combinatoric **187** (1978), 129 – 141.

[4] **S.E. Bammel and J. Rothstein**, *The number of* 9×9 *latin squares*, Discrete Math. **11** (1975), 83 – 95.

[5] **L.J. Brant and G.L. Mullen**, *A note on isomorphism classes of reduced latin squares of order* 7, Utilitas Mathematica **27** (1985), 261 – 263.

[6] **J.W. Brown**, *Enumeration of latin squares with application to order* 8, J. Combinatorial Theory **5** (1972), 177 – 184.

[7] **A. Cayley**, *On latin squares*, Oxford Camb. Dublin Messenger of Math. **19** (1890), 85 – 239.

[8] **D. Daly and P. Vojtechovsky**, *Enumeration of nilpotent loops via cohomology*, J. Algebra 322 (2009), 4080 – 4098.

[9] **G. Dequen and O. Dubois**, *The non-existence of a* $(3,1,2)$*-conjugate orthogonal Latin square of order* 10, In: Principles and Practice of Constraint Programming (CP), 108 – 201.

[10] **L. Euler**, *Recherches sur une nouvelle espéce de quarrés magiques combinatorial aspects of relations*, Verhandelingen uitgegeven door het zeeuwsch Genootschap der Wetenschappen de Vlissingen **9** (1782), 85 – 239.

[11] **R.A. Fisher and F. Yates**, *The* 6×6 *latin squares*, Proc. Cambridge Philos. Soc. **30** (1934), 492 – 507.

[12] **M. Frolov**, *Sur les permutations carrées*, J. de Math. spéc IV (1890), 8 – 11, 25 – 30.

[13] **I. Gessel**, *Counting latin rectangles*, Bull. Amer. Math. Soc. **16** (1987), 79 – 83.

[14] **A. Hulpke, P. Kaski and P.R.J. Östergård**, *The number of Latin squares of order* 11, Math. Comp. 80 (2011), 1197 – 1219.

[15] **S.M. Jacob**, *The enumeration of the latin rectangle of depth three by means of a formula of reduction, with other theorems relating to non-clashing substitutions and latin squares*, Proc. London Math. Soc. **31** (1930), 329 – 354.

[16] **M. Kinyon and I.M. Wanless**, *Loops with exponent three in all isotopes*, Internat. J. Algebra Comput., **25** (2015), 1159 – 1177.

[17] **G. Kolesova, C.W.H. Lam, and L. Thiel**, *On the number of* 8×8 *latin squares*, J. Combin. Theory Ser. A **54** (1990), 143 – 148.

[18] **P.A. MacMahon**, *Combinatory Analysis*, Cambridge University Press **1** (1915).

[19] **R. Maddux**, *Some varieties containing relation algebras*, Trans. Amer. Math. Soc. **272** (1982), 501 – 526.

[20] **B.D. McKay, A. Meynert, and W. Myrvold**, *Small latin squares, quasigroups, and loops*, J. Combinatorial Designs **15** (2007), 98 – 119.

[21] **B.D. McKay and E. Rogoyski**, *Latin squares of order* 10, Electron. J. Combin. **2** (1995), R3, 1 – 4.

[22] **B.D. McKay and I.M. Wanless**, *On the number of latin squares*, Ann. Combin. **9** (2005), 335 – 344.

[23] **J.R. Nechvatal**, *Asymptotic Enumeration of Generalised Latin Rectangles*, Util. Math. **20** (1981), 273 – 292.

[24] **H.W. Norton**, *The* 7×7 *squares*, Ann. Eugenics **9** (1939), 269 – 307.

[25] **QSCGZ. (pseudonym)**, *Anonymous electronic posting to Loopforum*, (2001). http://groups.yahoo.com/group/loopforum.

[26] **A. Sade**, *An omission in norton's list of* 7×7 *squares*, Ann. Math. Stat. **22** (1951), $306 - 307$.

[27] **A. Sade**, *Morphismes de quasigroupes: Tables*, Revista da Faculdade de Ciências de Lisboa, 2: A – Ciências Matemáticas **13** (1970/71), $149 - 172$.

[28] **P.N. Saxena**, *A simplified method of enumerating latin squares by MacMahon's differential operators; ii. the* 7×7 *latin squares*, J. Indian Soc. Agric. Statistics **3** (1951), $24 - 79$.

[29] **E. Schönhardt**, *Über lateinische quadrate und unionen*, J. für die reine und angewandte Mathematik **163** (1930), $183 - 230$.

[30] **M.C. Slattery and A.L. Zenisek**, *Moufang loops of order* 243, Comment. Math. Univ. Carolin. **53** (2012), $423 - 428$.

[31] **J. Slayney and A. Ali**, *Generating loops with the inverse property*, Proc. of ESARM, (2008), $55 - 66$.

[32] **G. Tarry**, *Le probléme des* 36 *officiers*, Ass. Franc. Paris **29** (1900), $170 - 203$.

[33] **M.B. Wells**, *The number of latin squares of order eight*, J. Combinatorial Theory **3** (1967), $98 - 99$.

Received October 30, 2016

College of Computer Engineering and Science
Prince Mohammad Bin Fahd University
Al-Khobar, 31952
KSA
e-mail: makhan@pmu.edu.sa, smuhammad@pmu.edu.sa, nmohammad@pmu.edu.sa

Faculty of Mathematics Department
Quaid-e-Azam University
Islamabad
Pakistan
e-mail: dr_asif_ali@hotmail.com