Genetic Search for Quasigroups with Heterogeneous Power Sequences

Eliška Ochodková, Pavel Krömer, Jiří Dvorský, Jan Platoš, Ajith Abraham, Václav Snášel
Department of Computer Science
FEECS, VŠB Technical University of Ostrava
Ostrava, Czech Republic
Email: {eliska.ochodkova,pavel.kromer,jiri.dvorsky,jan.platos,ajith.abraham,vaclav.snasel}@vsb.cz

***Abstract*—Genetic algorithms have been successfully used for searching for quasigroups with good properties. In this study we extend previous work done on evolutionary search for quasigroups by defining new fitness functions based on selected algebraical properties of the quasigroups. Introduced fitness functions make use of heterogeneous sequences generated during the exponentiation of quasigroups elements. Effects of the artificial evolution utilizing proposed fitness functions are illustrated on quasigroups of the order 8, 10, and 12.**

***Keywords*-genetic algorithms, quasigroups, optimization, fitness**

I. Introduction

Quasigroups, equivalent to the more familiar Latin squares, are not only interesting theoretical objects or mathematical puns, but they also have their practical application, e.g. in cryptography. The necessity of using suitable quasigroups as basic algebraic structures of the new cryptographic algorithms has been mentioned in many articles, eg. [8], [11], [13], [19]. Among the number of minor proposals, there are also serious proposals of cryptographic algorithms based on quasigroups. The following can be mentioned as representatives: the stream cipher Edon80 [10], published as an eSTREAM[1] candidate, and the NIST's SHA-3[2] competition candidate, the hash function Edon$\mathcal{R}$ [9]. Their authors had to select and use the most appropriate from all existing quasigroups. If a quasigroup is the base of a cryptographic primitive, it is necessary to examine whether its algebraic properties or other features mean a security risk to the whole cryptographic algorithm. From all existing quasigroups of a given order, those that do not have various identities (as associativity is) or those in which these identities appear rarely, had to be selected. Such a quasigroup can be defined as e.g. shapeless quasigroup [8].

Properties of small quasigroups (e.g. of order 4), represented as a look-up table only, may be examined by the exhaustive search. Another situation occurs when working with quasigroups of a higher order, or, when testing certain properties of group is time-consuming. We propose genetic algorithms and three new fitness functions based on a product of sequence to search for a quasigroup, in this paper. A product of sequence is used to determine how associative a quasigroup is.

[1]http://www.ecrypt.eu.org/stream/
[2]http://csrc.nist.gov/groups/ST/hash/sha-3/

II. Background

In this section we summarize the necessary definitions.

Definition 1: A quasigroup is a pair $(Q, \circ)$, where $\circ$ is a binary operation on (finite) set Q such that for all not necessarily distinct $a, b \in Q$, the equations $a \circ x = b$ and $y \circ a = b$. have unique solutions.

The fact that the solutions are unique guarantees that no element occurs twice in any row or column of the table for $(\circ)$. However, in general, the operation $(\circ)$ is neither a commutative nor an associative operation. Among 576 distinct quasigroups of order 4 only 16 are associative. That means that non-associative quasigroups dominate heavily.

Quasigroups are equivalent to more familiar Latin squares. The multiplication table of a quasigroup of order q is a Latin square of order q, and vice versa, as it was indicated in [2], [3], [16], every Latin square of order q is the multiplication table of a quasigroup of order q.

Definition 2: Let $A = \{a_1, a_2, \ldots, a_n\}$ be a finite alphabet, a $n \times n$ Latin square L is a matrix with entries $l_{ij} \in A$, $i, j = 1, 2, \ldots, n$, such that each row and each column consists of different elements of A. n is the order of the Latin square.

For $i, j, k \in A$ the ordered triple $(i, j; k)$ is used to represent the occurrence of element k in the cell (i, j) of the Latin square. So a Latin square may be represented by the set $\{(i, j; k)|$ entry k occurs in cell (i, j) of the Latin square $L.\}$

All Latin squares of order n are enumerated for $n \leq 11$ in [12]. The problem of classification and exact enumeration of Latin squares of an order greater than 11 probably still remains unsolved. Number of distinct Latin squares of a given order grows exceedingly quickly with the order and there is no known easily-computable formula for the number of distinct Latin squares. Thus, there are more than 10^{90} quasigroups of order 16 and if we take an alphabet $L = \{0 \ldots 255\}$ (i.e. data are represented by 8 bits) there are at least $256!255! \ldots 2! > 10^{58000}$ quasigroups.

A. *Product of sequence*

Let $(\circ)$ be the binary operation. Let A be a finite sequence of elements $a_1, \ldots, a_n$, $a_i \in A, i = 1, 2, \ldots, n, n \geq 2$. What does a product of this sequence mean? Clearly, for $n = 2$ we have $a_1 \circ a_2$, by juxtaposition $a_1 a_2$. For $n = 3$ a product of the sequence a_1, a_2, a_3 is defined as a set consisting of product elements $a_1(a_2 a_3)$ and $(a_1 a_2)a_3$. The product is denoted as $\{a_1 a_2 a_3\}$ and symbol $a_1 a_2 a_3$ means any *product element*. Generally, we can define a product of a sequence of n elements of A as follows below [1].

Definition 3: The product of a sequence $a_1, a_2, \ldots, a_n$ of elements $a_i \in A, i = 1, 2, \ldots, n$ is the set $\{a_1 a_2 \ldots a_n\}$ defined as:

- for $n = 2$ the set $\{a_1 a_2\}$ consist of only one element $a_1 a_2$,
- for $n \geq 2$ the set $\{a_1 a_2 \ldots a_n\}$ is defined as

$\{a_1 a_2 \ldots a_n\} = \{a_1\}\{a_2 \ldots a_n\} \cup \{a_1 a_2\}\{a_3 \ldots a_n\} \cup \ldots \cup \{a_1 \ldots a_{n-1}\} \cup \{a_n\}$.

The n elements $a_i \in A$ can be joined (multiplied), without changing their order, in $\frac{(2n-2)!}{n!(n-1)!}$ ways. For e.g. $n = 1, 2 \ldots, 10$ we obtain $1, 1, 2, 5, 14, 42$, $132, 429$, $1430, 4862$ ways of joining n elements. These numbers are called *Catalan numbers* [7]. The $m-$th Catalan number, for $m \geq 0$ is given by:

$$C_m = \frac{1}{m+1}\binom{2m}{m} = \frac{(2m-2)!}{m!(m-1)!}.$$

If the associative property does not hold for operation $(\circ)$ on the set A, we can generally obtain distinct values $a_1 a_2 \ldots a_n$ (not one common value) for all C_n possible product elements of the product set $\{a_1 a_2 \ldots a_n\}$ of the sequence $a_1, a_2, \ldots, a_n$.

If all elements $a_i \in A$ are equal, each element is denoted as a, a product $\{\underbrace{aa\ldots a}_{k}\}$ of the sequence $\underbrace{a, a, \ldots, a}_{k}$ can be enumerated. This product consists of all C_k product elements. The question is, how many distinct values of power $\underbrace{aa\ldots a}_{k} = a^k$ for all $a \in Q$ can be obtained. In the ideal case, all possible values can be obtained; the set of possible values has only max. n values from Q (of order n) for all powers a^k.

Better information about the identities in the given quasigroup can be acquired from the evaluation of particular product elements by the transformation shown in Fig. 1. Therefore all sequences $b_1 \ldots b_k$ obtained during the evaluation of product elements (powers) a^k are important. For each $a \in Q$, C_k distinct sequences $b_1 \ldots b_k$ can be obtained (C_k is the k-th Catalan number, e.g. for $k = 8$, $C_8 = 429$).

A product of he tsequence compared to a number of associative triplets for certain quasigroups was also used to determine how associative quasigroups are in [19].

$$\begin{tikzcd}
 & a \arrow[d] & a \arrow[d] & \cdots & a \arrow[d] & a \arrow[d] \\
a \arrow[ur] & b_1 \arrow[ur] & b_2 \arrow[ur] & \cdots \arrow[ur] & b_7 \arrow[ur] & b_8 = a^8
\end{tikzcd}$$

Figure 1: Transformation used for valuing the product elements $a^k, k = 8$ quasigroups

B. *Isotopism of quasigroups*

Definition 4: Let $(G, \cdot)$, $(H, \circ)$ be two quasigroups. An ordered triple (π, ρ, ω) of bijections π, ρ, ω of the set G onto set H is called an *isotopism* of $(G, \cdot)$ upon $(H, \circ)$ if $\forall u, v \in G, \pi(u) \circ \rho(v) = \omega(u \cdot v)$. Quasigroups $(G, \cdot)$, $(H, \circ)$ are said to be *isotopic*.

We can imagine an isotopism of quasigroups as a permutation of rows and columns of a quasigroup's multiplication table.

Example 1: Consider a multiplication table for a quasigroup isotopic to the quasigroup of modular subtraction, with operation $(\circ)$ defined as $a \circ b = (a + n - b) \bmod n$

The following table was created from the quasigroup of modular subtraction. The second and the third row were exchanged. Permutations π, ρ were identities and $\omega = [0213]$. A multiplication in this quasigroup can be illustrated by e.g. $1 \circ 0 = \omega(1) \circ 0 = 2 \circ 0 = 2$.

$\circ$	0	1	2	3
0	0	3	2	1
1	2	1	0	3
2	1	0	3	2
3	3	2	1	0

Starting with a base quasigroup, e.g. the quasigroup of modular subtraction, we can explore the large class of quasigroups isotopic to the base quasigroup [5], [15]. The multiplication in such an isotopic quasigroup is defined as follows:

$$a \circ b = \pi^{-1}\left((\omega(a) + n - \rho(b)) \bmod \ n\right). \qquad (1)$$

The quasigroup defined by its multiplication formula without the multiplication table stored is called an *analytic quasigroup* [20]. In contrast, the quasigroup with multiplication table stored in computer memory will be through the rest of this paper called a *table quasigroup*.

C. *Creating isotopic quasigroups*

Isotopic quasigroups can be obtained easily for table quasigroups. Consider a quasigroup of the order n defined by multiplication $a \circ b = (a + n - b) \bmod n)$. Then the three permutations π, ρ, ω must be chosen in order to find isotopic quasigroup, whose multiplication will be defined by Eq. (1).

There is $n!$ different permutations of n elements. Because three independent permutations are used to define the isotopic quasigroup, there are $n!n!n!$ possible choices of the triple π, ρ and ω. The permutations of elements can be straightforwardly evaluated with the multiplication table in computer memory.

On the other hand, the permutations of elements cannot be performed so easily for an analytic quasigroup because its multiplication table is not available. Instead, the permutation needs to be implemented as a function of an element of Q. The genetic algorithm for quasigroup optimization implements this task using the bit permutation of elements [21].

A quasigroup over a set of n elements requires $\log_2(n)$ bits to express each element. Each permutation of bits in the element representation defines also a permutation of all elements of the quasigroup, if n is a power of 2. The bit permutation can be easily implemented as a function of $q \in Q$.

The bit permutation is an elegant way of implementing permutations over the n elements of Q. Although it allows us to explore only a subset $(\log_2(n)! \log_2(n)! \log_2(n)!)$ of all possible permutation triples,it is still useful, because it does not require all n elements in main memory and therefore it can be used to search for quasigroups isotopic to analytic base quasigroups.

Bit permutations are computationally more expensive than the static functions used to implement permutation in [22]. However, there are ongoing efforts to implement bit permutation in hardware, which would improve the performance of the proposed algorithm significantly [6]. The main disadvantage of bit permutations is the fact that the GA that uses them can browse only a small part of the space of all quasigroups isotopic to the selected base quasigroup and that the order of the quasigroup has to be power of 2 (e.g. 4, 8, 16 and so on). The quasigroups with multiplication table in main memory impose no such requirement and the genetic algorithm can be used to search for all quasigroups isotopic to table quasigroup of arbitrary order.

III. Genetic algorithms

Genetic algorithms (GAs) are among the most popular and most successful meta-heuristic optimization algorithms. The GAs seek for an optimal or sub-optimal solution of a given problem by mimicking the natural evolution. Candidate solutions are encoded into so called chromosomes, linear data structures that are suitable for application of the genetic operators. The genetic operators implement the principles of biological evolution in software. The most important GA operators are the crossover operator that simulates sexual reproduction of haploid organisms and the mutation operator introducing random modifications to the genotype. The whole process is designed to improve the goodness of the population of candidate solutions. The quality of each solution is evaluated using so called fitness function, a domain specific function that assigns higher score to better candidate solutions [14].

The basic workflow of the standard generational GA is shown in Fig. 2.

```
1 Define objective (fitness) function and problem encoding;
2 Encode initial population $P$ of possible solutions as fixed length
  strings;
3 Evaluate chromosomes in initial population using objective function;
4 while Termination criteria not satisfied do
5     Apply selection operator to select parent chromosomes for
      reproduction: $sel(P_i) \rightarrow parent1$, $sel(P_i) \rightarrow parent2$;
6     Apply crossover operator on parents with respect to crossover
      probability to produce new chromosomes:
      $cross(pC, parent1, parent2) \rightarrow$
      $\{offspring1, offspring2\}$;
7     Apply mutation operator on offspring chromosomes with respect
      to mutation probability:
      $mut(pM, offspring1) \rightarrow offspring1$,
      $mut(pM, offspring2) \rightarrow offspring2$;
8     Create new population from current population and offspring
      chromosomes:
      $migrate(offspring1, offsprig2, P_i) \rightarrow P_{i+1}$;
9 end
```

Figure 2: A summary of genetic algorithm

Many variants of the standard generational GA have been proposed. The differences are mostly in particular selection, crossover, mutation and replacement strategy [14].

IV. Genetic search for good quasigroups

Genetic algorithms were first applied to the quasigroup optimization in [21] and the concept was further extended in [17], [18]. The artificial evolution was used to search for good quasigroups isotopic to the quasigroup of modular subtraction. Because the bit permutation was used, large quasigroups could have been explored without extensive memory requirements.

The genetic algorithm for the search for analytic quasigroup is defined by encoding of the candidate solutions and fitness function to evaluate chromosomes. The encoding employed in this work is very simple; we represent each permutation using the random keys encoding and the permutation triple as a string of three concatenated permutations.

A. Fitness function based on products of sequences

A number of fitness functions was already used for quasigroup optimization. Fitness functions based on hashing [21], randomness of generated sequences [18], and the associativity and commutativity [17] were found to be able to drive the evolutionary optimization of isotopic quasigroups in given direction.

In this paper, we define new fitness functions based on a recently formulated property of quasigroups, the heterogeneity (randomness) of products of sequences [19]. We wish to find a quasigroup that would generate as heterogeneous products of k elements $a \in Q$ i.e. a^k as possible. Moreover, we want the quasigroup to generate different sequences of intermediate results $b_1 b_2 \dots b_k$ when computing a^k for all $a \in Q$ using all possible products elements. In this paper only the highest power $k = n$ is used to compute heterogeneity (randomness) of products of sequences.

Let's once again suppose quasigroup of modular subtraction $(Q, \circ)$, where $Q = \{0, 1, \ldots, n-1\}$ and operation $\circ$ is defined as $a \circ b = (a + n - b) \bmod n$. Consider a vector of results of the power operation $\vec{r} = (r_0, r_1, \ldots, r_{n-1})$, where r_i is the number of cases in which $a^n = i$ for all $a \in Q$ and all product elements of n elements of Q. Let $\vec{i} = (i_0, i_1 2, \ldots, i_{n-1})$ be a vector of intermediate results where i_i is the number of times $b_j = i$ during the evaluation of a^n for all $a \in Q$, $1 \leq j \leq n$ and all product elements. Let $\vec{s} = (s_0, s_1, \ldots, s_{n-1})$ is the vector containing the number of distinct sequences of intermediate results obtained when computing a^n, i.e. s_i equals to the number of all distinct sequences generated when computing i^n.

We define three fitness functions based on the product elements:

$$f_1^{a^n}\left(\vec{r}, \vec{i}\right) = \frac{1}{2}\left(\frac{\sum_{j, \vec{r}_j > 0}(1)}{n} + \frac{\sum_{j, \vec{i}_j > 0}(1)}{n}\right) \tag{2}$$

$$f_2^{a^n}\left(\vec{r}, \vec{i}\right) = \frac{1}{2}\left(\frac{\max(\vec{r}) - \min(\vec{r})}{\max(\vec{r})} + \frac{\max\left(\vec{i}\right) - \min\left(\vec{i}\right)}{\max\left(\vec{i}\right)}\right) \tag{3}$$

$$f_3^{a^n}(\vec{s}) = \frac{\max(\vec{s}) + \min(\vec{s})}{2C_n} \tag{4}$$

where C_n is the n-th Catalan number.

The fitness function $f_1^{a^n}$ assigns large fitness to quasigroups in which it holds: $\forall q \in Q\ \exists a : a^n = q$ and $\forall q \in Q\ \exists a \in Q\ \exists l \in \{1, 2, \ldots n\} : a^l = q$. In another words, it prefers quasigroups that generate such sequences that contain all elements of Q at least once. The fitness function $f_2^{a^n}$ assigns large fitness to quasigroups that generate sequences containing all elements of Q with approximately the same frequency. Finally, the third fitness function assigns large fitness to quasigroups that generate as different sequences of intermediate results as possible.

V. Experimental optimization

To evaluate the proposed fitness function, we were looking for a quasigroup isotopic to the quasigroup of modular subtraction of the order 8 which was expressed as both, the analytic quasigroup and table quasigroup. Then we were searching for quasigroups isotopic to the quasigroup of modular subtraction of orders 10 and 12 respectively. We did not explore larger quasigroups due to the size and complexity of the set of products of sequences that are necessary to generate and evaluate.

The associativity of the quasigroup of modular subtraction with orders 8, 10, and 12 is visualized in Fig. 3, Fig. 8, and Fig. 11 respectively.

To visualize a quasigroup a grey map is used. The grey map consists of $n \times n$ squares of some shade of grey color. The grey map visualizes the sequences of intermediate results $b_1 b_2 \ldots b_n$ in the following way: let's define a $n \times n$ matrix M and the elements of the matrix $M_{b_i, b_{i+1}}$ as the number of pairs $b_i b_{i+1}$ in all sequences of intermediate results $b_1 b_2 \ldots b_n$. Elements of the matrix M are then converted to shades of grey, where the color black represents 0 and white represents the maximal value of M. It is obvious that the ideal distribution of values in M is uniform, so the grey map consists of squares with the same shade of grey color. The more different shades of grey on the image, the less associative the quasigroup is. We can clearly see that the quasigroups have flaws, for example for 0^8 in Fig. 3.

A random quasigroup isotopic to the quasigroup of modular subtraction for $k = 8$ is shown in Fig. 4. It demonstrates that a random isotopic quasigroup can have *worse* properties than the base quasigroup. In this case, the powers of 0, 3, and 5 generate undesirable sequences.

The values of fitness functions for the base quasigroup, random isotopic quasigroup and optimized isotopic quasigroup are shown in table I. It shows that the fitness function $f_1^{a^n}$ was not useful - the algorithm could did not manage to improve the value of this fitness function. Moreover, the quasigroups found with $f_1^{a^n}$ generated bad sequences e.g. for 3^8. Because of this find, we did exclude $f_1^{a^n}$ from further experiments and from the visualization.

The value of the other two fitness functions was improved by the GA and hence they can be used to search for better isotopic quasigroups. In the rest of this section, we provide a visualization of selected quasigroups.

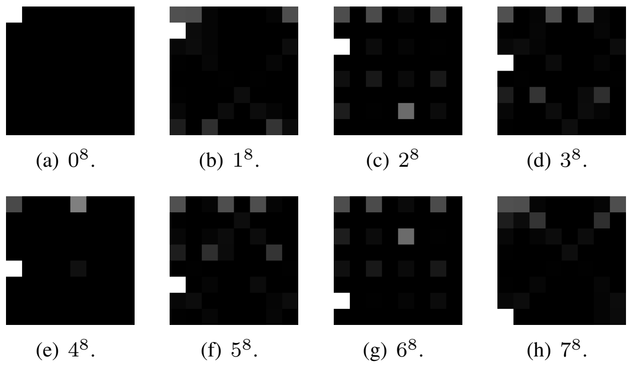
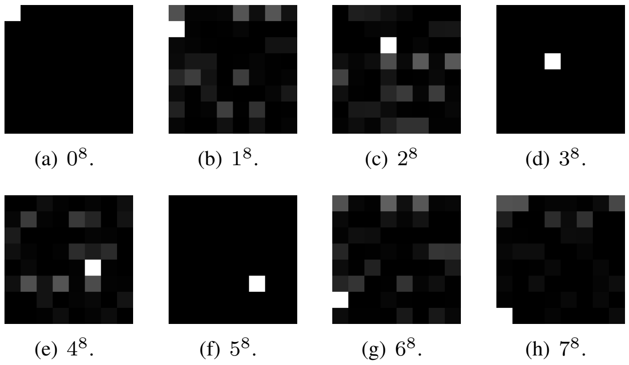


Figure 3: The quasigroup of modular subtraction with order 8.

The quasigroup found by the GA with bit mutation is shown in Fig. 5. We see that the optimized quasigroup has better associativity, but still generates wrong sequences for 0^8. This is due to the inability of GA with bit permutation to cover the space of all quasigroups isotopic to the base quasigroup.

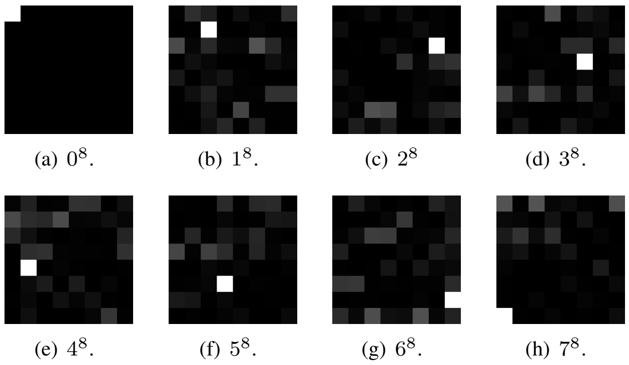
The Fig. 6 shows the results of genetic search using the table form of the quasigroup of modular subtraction. Such a search allows us to explore all isotopic quasigroups.



(a) 0^8. (b) 1^8. (c) 2^8 (d) 3^8.

(e) 4^8. (f) 5^8. (g) 6^8. (h) 7^8.

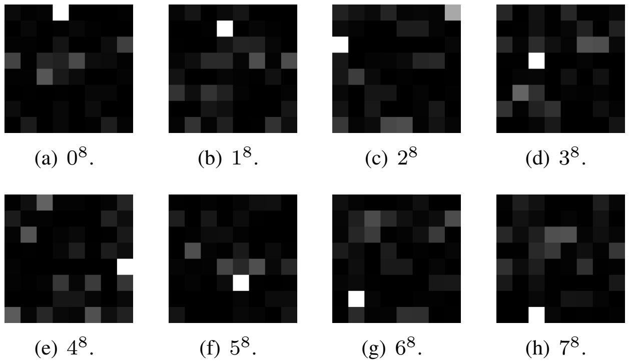
Figure 4: A random quasigroup isotopic to the quasigroup of modular subtraction with order 8.



(a) 0^8. (b) 1^8. (c) 2^8 (d) 3^8.

(e) 4^8. (f) 5^8. (g) 6^8. (h) 7^8.

Figure 5: The quasigroup found with $f_2^{a^n}$



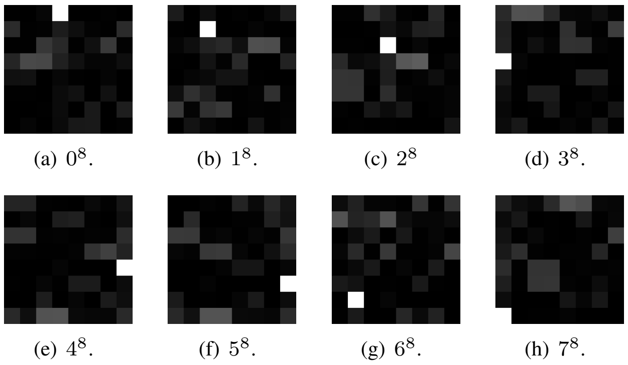
(a) 0^8. (b) 1^8. (c) 2^8 (d) 3^8.

(e) 4^8. (f) 5^8. (g) 6^8. (h) 7^8.

Figure 6: The quasigroup found with $f_2^{a^n}$ (table quasigroup used)

We can see that the optimized quasigroups have now no error for 0^8. The quasigroup found with $f_2^{a^n}$ has no such flaws.

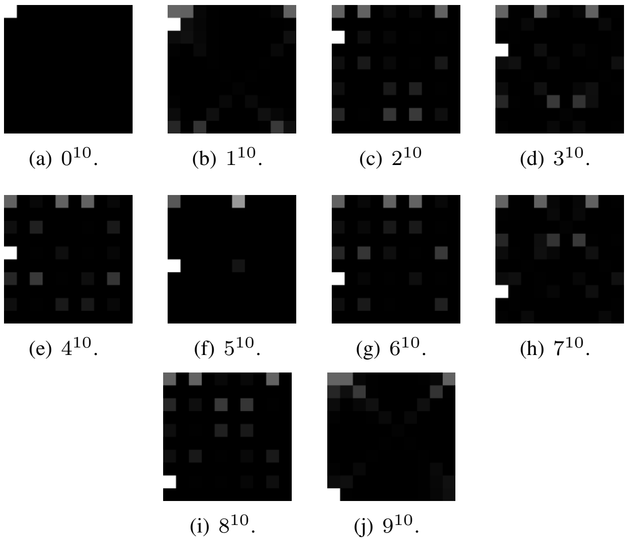
The GA was also used to find good isotopic table quasigroups with orders 10 and 12 shown in Fig. 9 and Fig. 12 respectively. We have used the fitness functions $f_2^{a^n}$ and $f_3^{a^n}$ because they put higher demands on the generated sequences and because quasigroups found by the GA with $f_1^{a^n}$ did contain errors for $k = 8$. Apparently, the quasigroups optimized by the GA have better associativity than base quasigroups, i.e. the quasigroups of modular subtraction of the orders 10 and 12.



(a) 0^8. (b) 1^8. (c) 2^8 (d) 3^8.

(e) 4^8. (f) 5^8. (g) 6^8. (h) 7^8.

Figure 7: The quasigroup found with $f_3^{a^n}$ (table quasigroup used)



(a) 0^{10}. (b) 1^{10}. (c) 2^{10} (d) 3^{10}.

(e) 4^{10}. (f) 5^{10}. (g) 6^{10}. (h) 7^{10}.

(i) 8^{10}. (j) 9^{10}.

Figure 8: The quasigroup of modular subtraction for $k = 10$

The parameters of the algorithm, see Table II, were selected after initial tuning of the algorithm, and they were selected according to the best practices, see [14].

Table I: Average fitness values of investigated quasigroups

k	fitness func.	quasigroup of mod. subtraction	rand. isotopic quasigroup	opt. isotopic quasigroup
8	$f_1^{a^n}$	0.9375	0.9375	0.9375
	$f_2^{a^n}$	0.416309	0.41466	0.47712
	$f_3^{a^n}$	0.600233	0.66434	0.8042
10	f_2^k	0.42292	0.46264	0.48352
	$f_3^{a^n}$	0.67719	0.69375	0.77283
12	f_2^k	0.46862	0.47579	0.49304
	$f_3^{a^n}$	0.4819	0.51042	0.74227

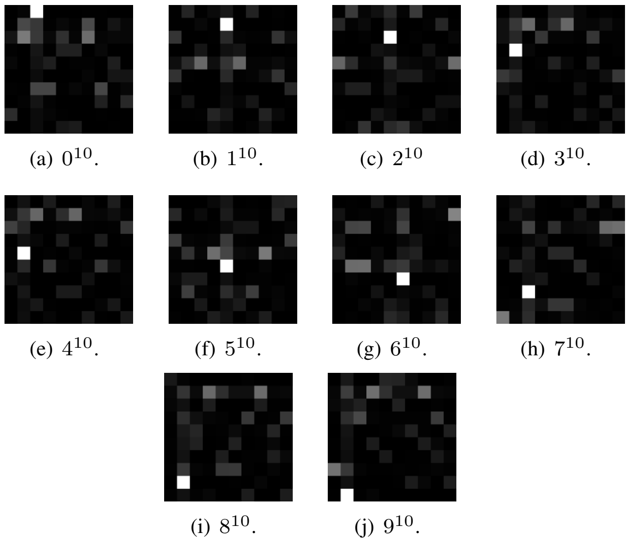


Figure 9: The quasigroup found with $f_2^{a^n}$ (table quasigroup used) $n = 10$

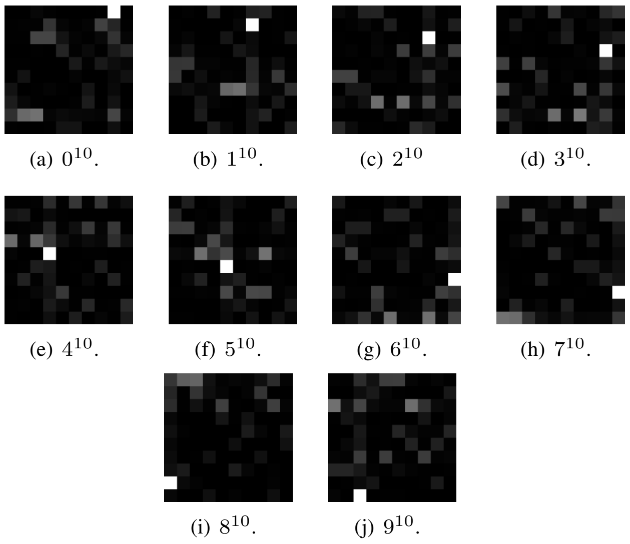


Figure 10: The quasigroup found with $f_3^{a^n}$ (table quasigroup used) $n = 10$

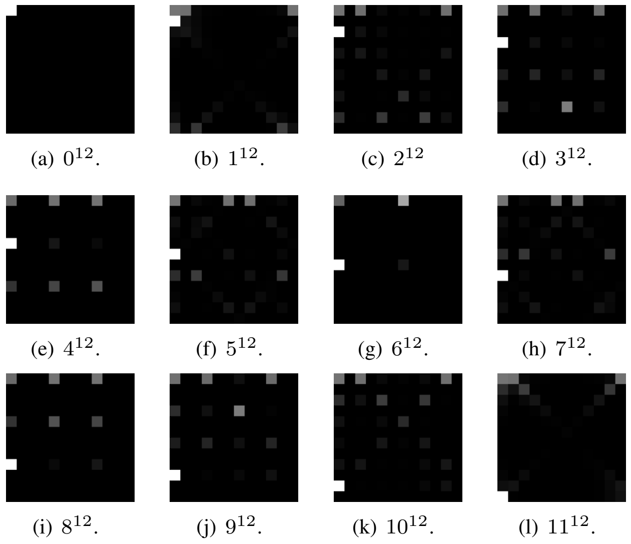


Figure 11: The quasigroup of modular subtraction for $k = 12$

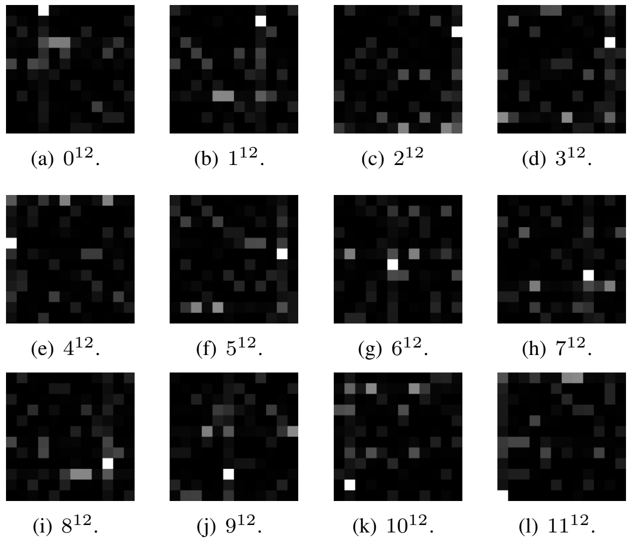


Figure 12: The quasigroup found by fitness func. $f_2^{a^n}$ (table quasigroup used) $n = 12$

VI. Conclusions

We have applied genetic algorithms to search for a quasigroup. The quasigroup was given either by a multiplication table or by an analytical definition of its operation. Three new fitness functions based on the products of sequences were defined and evaluated. The GA was able to find quasigroups with better (i.e. less) associativity than the base quasigroup when using two of the three proposed fitness functions.

Our future work will include a further study of fitness functions to measure the associativity of a quasigroup. Moreover, we will implement the evaluation of the fitness function in a parallel environment, e.g. on CUDA, to enable the application of proposed fitness functions to the quasigroups of greater order. Statistical properties of randomly generated quasigroups will be also tested.

Acknowledgment

This work was partially supported by the Czech Science Foundation (GAČR), grant No. 205/09/1079.

References

[1] O. Borůvka. *Foundations of the theory of groupoids and groups.* Wiley, 1976.

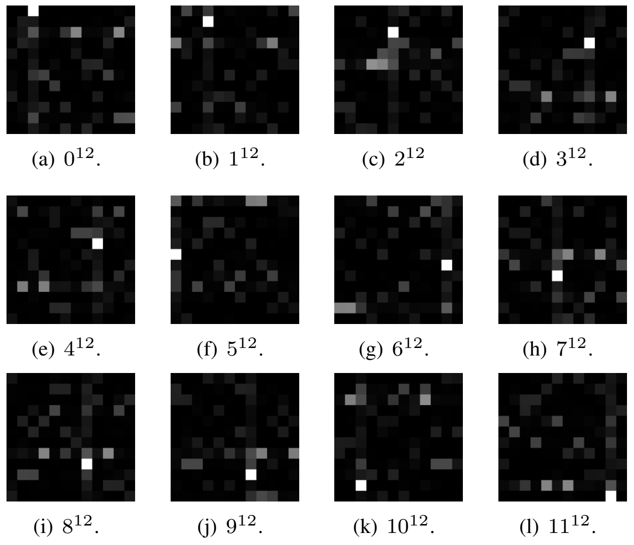

(a) 0^{12}. (b) 1^{12}. (c) 2^{12} (d) 3^{12}.
(e) 4^{12}. (f) 5^{12}. (g) 6^{12}. (h) 7^{12}.
(i) 8^{12}. (j) 9^{12}. (k) 10^{12}. (l) 11^{12}.

Figure 13: The quasigroup found by fitness func. $f_3^{a^n}$ (table quasigroup used) $n = 12$

Table II: The settings of genetic algorithm for quasigroup search

Parameter	Value
Fitness function	$f_1^{a^n}$, $f_2^{a^n}$, $f_3^{a^n}$
Quasigroup order n	8, 10, 12
Population size	10
Probability of mutation P_M	0.2
Probability of recombination P_C	0.8
Selection	roulette wheel selection
Max number of generations	2000

[2] Belousov, V. D. Osnovi teorii kvazigrup i lup (in Russian), Nauka, Moscow, 1967.

[3] Dénes, J., Keedwell, A. Latin Squares and their Applications. Akadémiai Kiadó, Budapest; Academic Press, New York (1974)

[4] Dénes, J., Keedwell, A. A new authentication scheme based on Latin squares. Discrete Mathematics (106/107) (1992) pp. 157–161

[5] J. Dvorský, E. Ochodková, V. Snášel, Hash Functions Based on Large Quasigroups, Proceedings of Velikonoční kryptologie, Brno, 2002, pp. 1–8.

[6] Y. Hilewitz, Z. J. Shi, Lee, and R. B., "Comparing fast implementations of bit permutation instructions," in *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, California, USA), pp. 1856–1863, Nov. 2004 2004.

[7] P. Hilton, and J. Pedersen. Catalan Numbers, Their Generalization, and Their Uses. *Journal The Mathematical Intelligencer,* 13, no. 2 (1991): 64–75.

[8] D. Gligoroski, S. Markovski, L. Kocarev. EdonR, An Infinite Family of Cryptographic Hash Functions. *International Journal of Network Security,* 8 (3/2009): 293–300.

[9] D. Gligoroski, et al. EdonR cryptographic hash function. Submition to NIST's SHA-3 hash function competition, 2008, http://csrc.nist.gov/groups/ST/hash/sha-3/index.html

[10] D. Gligoroski, S. Markovski, L. Kocarev and J. Svein. The Stream Cipher Edon80. The eSTREAM Finalists, *Lecture Notes in Computer Science*, Vol. 4986. pp. 152-169 , 2008

[11] D. Gligoroski. Candidate One-Way Functions and One-Way Permutations Based on Quasigroup String Transformations. *Cryptology ePrint Archive.* Report 2005/352.

[12] B. D. McKay and I. M. Wanless. On the Number of Latin Squares. *Journal Annals of Combinatorics*, Issue Vol.ume 9 (2005), No. 3, pp. 335-344.

[13] S. Markovski, D. Gligoroski, and J. Markovski. Classification of quasigroups by random walk on torus. *Journal of Applied Mathematics and Computing* 19, no. 1-2 (2005): 57–75.

[14] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

[15] E. Ochodková, V. Snášel, Using Quasigroups for Secure Encoding of File System, Proceedings of the International Scientific NATO PfP/PWP Conference "Security and Information Protection 2001", May 9-11, 2001, Brno, Czech Republic, pp.175–181.

[16] J. D. H. Smith, An introduction to quasigroups and their representations, Chapman & Hall/CRC, 2007.

[17] V. Snášel, J. Dvorský, E. Ochodková, P. Krömer, J. Platoš, and A. Abraham, "Evolving quasigroups by genetic algorithms," in *DATESO*, vol. 567 of *CEUR Workshop Proceedings*, pp. 108–117, 2010. ISSN 1613-0073.

[18] V. Snášel, J. Dvorský, E. Ochodková, P. Krömer, J. Platos, and A. Abraham, "Genetic algorithms evolving quasigroups with good pseudorandom properties," in *ICCSA (3)* , vol. 6018 of *Lecture Notes in Computer Science*, pp. 472–482, Springer, 2010.

[19] E. Ochodková, J. Dvorský, V. Snášel, and A. Abraham, "Testing quasigroup identities using product of sequence," in *DATESO* , vol. 567 of *CEUR Workshop Proceedings*, pp. 155–162, CEUR-WS.org, 2010.

[20] V. Snášel, A. Abraham, J. Dvorský, P. Krömer, and J. Platoš, "Hash functions based on large quasigroups.," in *ICCS '09: Proceedings of the 9th Int. Conf. on Computational Science* , vol. 5544 of *Lecture Notes in Computer Science*, pp. 521–529, Springer, 2009.

[21] V. Snášel, A. Abraham, J. Dvorský, E. Ochodková, J. Platoš, and P. Krömer, "Searching for quasigroups for hash functions with genetic algorithms," in *World Congress on Nature & Biologically Inspired Computing, 2009. NaBIC 2009.*, pp. 367 – 372, IEEE Computer Society, 2009.

[22] V. Snášel, A. Abraham, J. Dvorský, P. Krömer, and J. Platoš, "Hash functions based on large quasigroups.," in *ICCS (1)* , vol. 5544 of *Lecture Notes in Computer Science*, pp. 521–529, Springer, 2009.