2006

Genetic Algorithm for Orthogonal Designs

Jason Cousineau
Wilfrid Laurier University

Ilias S. Kotsireas
Wilfrid Laurier University, ikotsire@wlu.ca

Christos Koukouvinos
National Technical University of Athens

AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **35** (2006), Pages 263–272

Genetic algorithms for orthogonal designs

Jason Cousineau* Ilias S. Kotsireas*

Wilfrid Laurier University
Department of Physics and Computer Science
75 University Avenue West
Waterloo, Ontario N2L 3C5
Canada

Christos Koukouvinos

Department of Mathematics
National Technical University of Athens
Zografou 15773, Athens
Greece

Abstract

We show how to use Simple Genetic Algorithm to produce Hadamard matrices of large orders, from the full orthogonal design or order 16 with 9 variables, $OD(16; 1, 1, 2, 2, 2, 2, 2, 2, 2)$. The objective function that we use in our implementation of Simple Genetic Algorithm, comes from a Computational Algebra formalism of the full orthogonal design equations. In particular, we constructed Hadamard matrices of orders 144, 176, 208, 240, 272, 304 and 336, from the aforementioned orthogonal design. By varying three genetic operator parameters, we computed 62 inequivalent Hadamard matrices of order 304 and 4 inequivalent Hadamard matrices of order 336. Therefore we established two new constructive lower bounds for the numbers of Hadamard matrices of orders 304 and 336.

1 Introduction

Definition Let $x_1, \ldots, x_t$ be commuting indeterminates. An orthogonal design X of order n and type $(s_1, \ldots, s_t)$ denoted $\mathrm{OD}(n; s_1, \ldots, s_t)$, where $s_1, \ldots, s_t$ are positive integers, is a matrix of order n with entries from $\{0, \pm x_1, \ldots, \pm x_t\}$, such that

$$XX^t = \left(\sum_{i=1}^{t} s_i x_i^2 \right) I_n,$$

* Supported in part by a grant from NSERC.

where X^t denoted the transpose of X and I_n denotes the identity matrix of order n. Orthogonal designs are used in Combinatorics, Statistics, Coding Theory, Telecommunications and other areas. For more details on orthogonal designs see the book [3] and the survey paper [9].

This paper is organized as follows. First we give a brief account of the construction of the full orthogonal design of order 16 with 9 variables, $OD(16;1,1,2,2,2,2,2,2,2)$ as well as the computational results obtained from it, in [8]. Then we describe Simple Genetic Algorithm and its application to the problem of computing Hadamard matrices of large order, using this orthogonal design.

Our implementation of SGA, using the objective functions that arise from the $OD(16;1,1,2,2,2,2,2,2,2)$ orthogonal design, was developed in the C programming language. The programs were executed on a Canadian supercomputer at SHARCnet, `www.sharcnet.ca` on Itanium2 900MHz and AMD Opteron 2.4GHz processors.

2 The full orthogonal design $OD(16;1,1,2,2,2,2,2,2,2)$

In [8], the authors use the algebra of sedenions and Gröbner bases to construct the full orthogonal design of order 16 with 9 variables $OD(16;1,1,2,2,2,2,2,2,2)$

$$OD_{16} = \begin{bmatrix}
A & B & C & D & E & F & G & H & I & B & C & D & E & F & G & H \\
B & A & D & C & F & E & H & G & B & I & D & C & F & E & H & G \\
C & D & A & B & G & H & E & F & C & D & I & B & G & H & E & F \\
D & C & B & A & H & G & F & E & D & C & B & I & H & G & F & E \\
E & F & G & H & A & B & C & D & E & F & G & H & I & B & C & D \\
F & E & H & G & B & A & D & C & F & E & H & G & B & I & D & C \\
G & H & E & F & C & D & A & B & G & H & E & F & C & D & I & B \\
H & G & F & E & D & C & B & A & H & G & F & E & D & C & B & I \\
I & B & C & D & E & F & G & H & A & B & C & D & E & F & G & H \\
B & I & D & C & F & E & H & G & B & A & D & C & F & E & H & G \\
C & D & I & B & G & H & E & F & C & D & A & B & G & H & E & F \\
D & C & B & I & H & G & F & E & D & C & B & A & H & G & F & E \\
E & F & G & H & I & B & C & D & E & F & G & H & A & B & C & D \\
F & E & H & G & B & I & D & C & F & E & H & G & B & A & D & C \\
G & H & E & F & C & D & I & B & G & H & E & F & C & D & A & B \\
H & G & F & E & D & C & B & I & H & G & F & E & D & C & B & A
\end{bmatrix} \tag{1}$$

This design is one of the full orthogonal designs of order 16 with 9 variables and appears in [3]. In [8], the authors discovered this design with Computational Algebra methods, see [2]. The complete list of such designs is given in [3].

Some new orthogonal designs for of orders 32 and 40, have recently been constructed using exhaustive computer searches in [7].

If we think of the 9 variables A, B, C, D, E, F, G, H, I, as numbers, then we have the relation

$$OD_{16}OD_{16}^t = (A^2 + 2B^2 + 2C^2 + 2D^2 + 2E^2 + 2F^2 + 2G^2 + 2H^2 + I^2)I_{16}$$

where the superscript t denotes matrix transposition and I_{16} stands for the unit matrix of order 16.

The array OD_{16} can be used to produce structured Hadamard matrices of orders $16 \times n$, where n is the order of the block matrices A, B, C, D, E, F, G, H, I. In the classical Williamson construction, the four matrices A, B, C, D are taken to be symmetric and circulant, see [5]. Imitating the classical Williamson construction, we take the nine matrices A, B, C, D, E, F, G, H, I to be symmetric circulant matrices of order n each, defined via the matrix U:

$$U = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 0 \end{pmatrix}.$$

which has the property $U^n = I_n$. Take the nine matrices A, B, C, D, E, F, G, H, I to be polynomials in U, so that they commute with each other:

$$\begin{array}{rcccccccc} A & = & a_0 I_n & + & a_1 U & + & \cdots & + & a_{n-1} U^{n-1} \\ B & = & b_0 I_n & + & b_1 U & + & \cdots & + & b_{n-1} U^{n-1} \\ C & = & c_0 I_n & + & c_1 U & + & \cdots & + & c_{n-1} U^{n-1} \\ D & = & d_0 I_n & + & d_1 U & + & \cdots & + & d_{n-1} U^{n-1} \\ E & = & e_0 I_n & + & e_1 U & + & \cdots & + & e_{n-1} U^{n-1} \\ F & = & f_0 I_n & + & f_1 U & + & \cdots & + & f_{n-1} U^{n-1} \\ G & = & g_0 I_n & + & g_1 U & + & \cdots & + & g_{n-1} U^{n-1} \\ H & = & h_0 I_n & + & h_1 U & + & \cdots & + & h_{n-1} U^{n-1} \\ I & = & i_0 I_n & + & i_1 U & + & \cdots & + & i_{n-1} U^{n-1} \end{array}$$

Since $U^T = U^{-1}$, the nine matrices A, B, C, D, E, F, G, H, I will be symmetric if

$$a_{n-i} = a_i, b_{n-i} = b_i, c_{n-i} = c_i, d_{n-i} = d_i,$$

$$e_{n-i} = e_i, f_{n-i} = f_i, g_{n-i} = g_i, h_{n-i} = h_i, i_{n-i} = i_i.$$

for $i = 1, \dots, n-1$.
When n takes specific values, the requirement

$$OD_{16} OD_{16}^t = (A^2 + B^2 + C^2 + D^2 + E^2 + F^2 + G^2 + H^2 + I^2) \odot I_{16}$$

boils down into systems of polynomial equations. Note that now we are using the Kronecker product $\odot$, because A, B, C, D, E, F, G, H, I are matrices of order n. Using these polynomial equations the authors in [8] conducted:

- exhaustive searches for $n = 3, 5, 7$ using automatically generated serial C programs;
- partial searches for $n = 9, 11$ using the WestGrid supercomputer Lattice, based at the University of Calgary.

Subsequently, in [8], the results of the searches for $n = 7$, 9 and 11 were used to locate sets of new inequivalent Hadamard matrices of orders 112, 144 and 176, respectively. These results establish new constructive lower bounds for the numbers of inequivalent Hadamard matrices of orders 112, 144 and 176.

The case $n = 13$, which corresponds to Hadamard matrices of order $16 \cdot 13 = 208$, proved to be a relatively hard computational challenge for the current state of the methods of [8]. In the following sections we show how to use Simple Genetic Algorithm to find solutions for $n = 13$, $n = 15$, $n = 17$, $n = 19$ and $n = 21$. We also find solutions for two smaller values of the parameter, namely for $n = 9$ and $n = 11$.

3 Simple Genetic Algorithm

In this section we summarize the concepts necessary to describe the Simple Genetic Algorithm (SGA) following the description in [4].
Genetic Algorithms were introduced in 1970 by John Holland [6] aiming to design an artificial system having properties similar to natural systems. Genetic algorithms can be applied to a variety of optimization and searching problems and work based on the concept of "survival of the fittest".
The three main ingredients of a general Genetic Algorithm are:

- a coding of the parameter set, usually in the form of a collection (or population in Biology) of binary vectors (individuals or chromosomes in Biology);
- an objective function, (or fitness function in Biology) that is to be minimized of maximized. In the sequel, the term objective function will be abbreviated as OF.
- a set of genetic operators, which are algorithmic analogs of biological processes in the Theory of Evolution.

It is customary to use the algorithmic and the biological terminologies interchangeably.
A Genetic Algorithm works by starting with a population of individuals chosen randomly. This population of individuals (or chromosomes) is initially generated at random. Each individual is a sequence of alleles. In our case the alleles are simply -1 and $+1$, which are coded as $-$ and $+$ respectively.

The value of the objective function is computed for each individual in the population. A fitness value is thus assigned to each individual in the population. The choice of the OF is crucial for the successful application of a Genetic Algorithm to a particular problem. Strings will then be selected to enter a mating pool. The probability of any string entering the pool is proportional to its fitness value. Many copies of strings with high fitness values may enter the pool, while relatively few strings with low fitness values will be selected. According to their fitness values, the most highly fit individuals are paired and genetic operators are applied to them. This gives rise to a new generation of individuals, the offspring. The value of the

objective function is again computed for each individual in the new generation. The values of the objective function in the new generation are expected to be better than the values of the objective function in the previous generation. The word "better" is interpreted as smaller or bigger, according to whether our aim is to minimize or maximize the objective function.
The Simple Genetic Algorithm (SGA), see [4], is a Genetic Algorithm in which we apply the three genetic operators of reproduction, crossover and mutation. We give a description of these three operators.

- **reproduction** stipulates that individuals in the population with higher objective function values (in the case where we maximize the objective function) must be attributed a higher probability of contributing offspring in the next generation. This genetic operator is an algorithmic analog of natural selection in the theory of evolution. The reproduction operator is often implemented in a computer program by a biased roulette. The result of the reproduction operator is a mating pool, which contains the individuals of the new generation.
- **crossover** acts on the individuals in the mating pool (the new generation) in two steps. First, these individuals are mated randomly into pairs. Second, each pair undergoes crossover by selecting a crossover site k randomly. This means that elements before and after the crossover site k are mutually exchanged. For instance if the two individuals $1, 1, 1, 1$ and $-1, -1, -1, -1$ have been mated and they will undergo crossover at the site $k = 3$, then the resulting individuals will be $1, 1, 1, -1$ and $-1, -1, -1, 1$. This genetic operator has many other variations, for example we can have two crossover sites. Crossover is generally not performed on every string, but instead occurs with a certain probability that is specified according to the particular problem.
- **mutation** changes randomly a bit from -1 to 1 or from 1 to -1, according to a certain probability. The mutation probability is often determined experimentally. The effects of this genetic operator are not entirely understood.

4 Results from Simple Genetic Algorithm

In using the Genetic Algorithm to find Hadamard matrices, we need to find strings of ± 1 values that give solutions to certain large systems of polynomial equations. These systems encode the Hadamard property $HH^t = hI$ for some particular order h. Two nice references for recent results concerning Hadamard matrices are [1] and [10].

In our C implementation of SGA, these strings are simply arrays of integers such that each element is either a $+1$ or a -1. A string's fitness is based on how many of the polynomials it solves. More specifically, the objective function is chosen as the sum of the absolute values of the equations of the polynomial system. Here are two examples of objective functions, for the OD16 array:

• $n = 3$

$$| a_0a_1 + 2b_0b_1 + 2c_0c_1 + 2d_0d_1 + 2e_0e_1 + 2f_0f_1 + 2g_0g_1 + 2h_0h_1 + i_0i_1 + 8 |$$

• $n = 5$

$$\begin{aligned} &| a_0a_2 + a_1a_2 + 2b_0b_2 + 2b_1b_2 + 2c_0c_2 + 2c_1c_2 + 2d_0d_2 + 2d_1d_2 + 2e_0e_2 + 2e_1e_2 \\ &\quad + 2f_0f_2 + 2f_1f_2 + 2g_0g_2 + 2g_1g_2 + 2h_0h_2 + 2h_1h_2 + i_0i_2 + i_1i_2 + 8 | + \\ &| a_0a_1 + a_1a_2 + 2b_0b_1 + 2b_1b_2 + 2c_0c_1 + 2c_1c_2 + 2d_0d_1 + 2d_1d_2 + 2e_0e_1 + 2e_1e_2 \\ &\quad + 2f_0f_1 + 2f_1f_2 + 2g_0g_1 + 2g_1g_2 + 2h_0h_1 + 2h_1h_2 + i_0i_1 + i_1i_2 + 8 | \end{aligned}$$

Extensive experimentations by the authors in using Genetic Algorithms in the OD16 array context, showed that a probability of crossover equal to 0.01 and a probability of mutation equal to 0.002 produced good results when used to find Hadamard matrices with order of approximately 200, and these values were also sufficient when searching for smaller matrices. In all cases, the population size was kept at 500,000 creatures so that the program would require no more RAM than the machines were equipped with.

4.1 Results for the OD16 array

In this paragraph, we mention the results we obtained via our C implementation of SGA, using the OD16 array to construct Hadamard matrices of order $16n$, where n is the order of the block matrices A, B, C, D, E, F, G, H, I. We used SGA for seven different values of n, in particular $n = 9$, 11, 13, 15, 17, 19 and 21 to construct Hadamard matrices of orders 144, 176, 208, 240, 272, 304 and 336 respectively.
For each odd value of n we need $\frac{9n+9}{2}$ variables to define the associated polynomial system. More specifically, we need $\frac{n+1}{2}$ variables to define each one of the 9 matrices A, B, C, D, E, F, G, H, I, via the polynomials in U. As a consequence, the objective function will be a function of $\frac{9n+9}{2}$ variables. The following table summarizes the parameter values, the orders of the Hadamard matrices and the number of variables required in each case.

n	9	11	13	15	17	19	21
order of Hadamard matrix	144	176	208	240	272	304	336
number of variables	45	54	63	72	81	90	99

In the presentation of the results we obtained obtained with SGA below, the solutions (which are sequences of $\frac{9n+9}{2}$ ± 1 variables) are given in the format:

$$a_0 \dots a_m b_0 \dots b_m c_0 \dots c_m d_0 \dots d_m e_0 \dots e_m f_0 \dots f_m g_0 \dots g_m h_0 \dots h_m i_0 \dots i_m$$

where $m = \frac{n-1}{2}$. For each solution found, we also give the number of generations evolved by SGA, the program execution time and the memory used by the program, as measures of the time and space complexity of the problem.

```
n = 9
45 variable OF:
11-1---1-1111-1-111----11---1-1-1----11-11-1-
Generation is 3, 45.18 seconds of execution time, 220 MB of memory.

n = 11
54 variable OF:
1-1--1-11----1---11--1-1-1---1-111---1----1--1111-1--1
Generation is 12, 126.93 seconds of execution time, 250 MB of memory.

n=13
63 variable OF:
1-11--1111---11--1-11--1-1--1--111111-11---11111-11-1-1-1-11--1
Generation is 2481, 249.95 seconds of execution time, 265 MB of memory.

n=15
72 variable OF:
1------111---1-111-1-1-1-1---11-11--11-11111-1---11-1--1-1111---1------1
Generation is 541, 6016.92 seconds of execution time, 287 MB of memory.

n=17
81 variable OF:
1--111-1--11---1----1-1--1111---11-1-111-1---------1-1-1-1----1--11-11
                                                                ---11---1-1
Generation is 4485, 58481.14 seconds of execution time, 322 MB of memory.

n=19
90 variables OF:
-1-1--111-1---1111--1--1---11---1--111--1--1--1-11---1-111111--11-1-111
                                                      ---1-1---1-1-11---1
Generation is 1725, 34589.37 seconds of execution time, 455 MB of memory.

n=21
99 variables OF:
------1--1111---11-111-1-1--1--1-1-1111--1----1111---1--1--111-11-111
                                           ---1--11-1---1-1-1-111111-11--
Generation is 46357, 90305.58 seconds of execution time, 406 MB of memory.
```

The fluctuations of the OF values and of the average OF value in each generation during the execution of SGA are useful parameters of the problem. To illustrate the behavior of OF values during SGA, we plotted these data for the case of the 72-variable OF for $n = 15$ for 541 generations. In this diagram, the upper line is the average fitness of all the strings, and the lower line is the fitness of the best string, which eventually becomes 0.

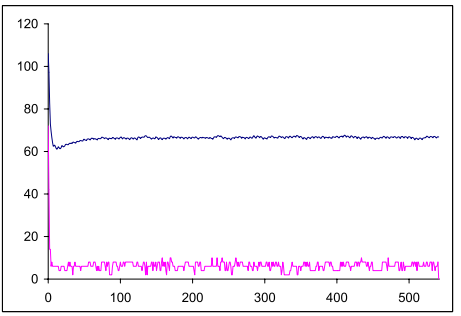

Figure 1: Evolution of the 72-variable OF values for $n = 15$.

4.2 New constructive lower bounds for the numbers of inequivalent Hadamard matrices of orders 304 and 336

In this paragraph, we give two new constructive lower bounds for the numbers of inequivalent Hadamard matrices of orders 304 and 336.
Notation: Let N_k denote the number of inequivalent Hadamard matrices of order k. We establish the inequalities:

$$N_{304} \geq 62, \qquad N_{336} \geq 4.$$

4.2.1 A new constructive lower bound for Hadamard matrices of order 304

In this paragraph we prove a new lower bound for Hadamard matrices of order 304, using our implementation of Simple Genetic Algorithm and Magma. We run SGA with the 90-variable OF for $n = 19$ and we computed 62 different solutions, listed in Appendix A. Subsequently, we computed with Magma V2.11-2 the 4-profiles of the 62 Hadamard matrices specified by these solutions, via the OD_{16} array. All 62 profiles turned out to be different and therefore these 62 matrices are inequivalent. The Maple code to verify the 62 solutions to the OF for $n = 19$ and the Magma code to compute the 62 profiles of the corresponding Hadamard matrices of order 304 are given in the web page `http://www.cargo.wlu.ca/ODGA-AJC/`. This web page also contains the explicit expression of the OF for $n = 19$, as well as the list of the 62 profiles.

4.2.2 A new constructive lower bound for Hadamard matrices of order 336

In this paragraph we prove a new lower bound for Hadamard matrices of order 336, using our implementation of Simple Genetic Algorithm and Magma. We run SGA with the 99-variable OF for $n = 21$ and we computed 4 different solutions, listed below:

```
------1--1111---11-111-1-1--1--1-1-1111--1----1111---1--1--111-11-111---
                                                1--11-1---1-1-1-111111-11--

1-1-1-1111-1----1-11--11111--1--1----1-111-11-11-----1--11-111---111--1-
                                                ---11--11-1-1--11-1-1-1111-

1--1--11-11-111--1-1--1----1-11-1-1--1111---11---11-1111-1---1-1---1-1--
                                                11--1111---11111-11-11--1--

--1--1---11-1--11--111-1-1-1---11-11111--11--1--1--1-1----1111111--11-11
                                                ----11----1-1-11--1--1---11
```

Subsequently, we computed with Magma V2.11-2 the 4-profiles of the 4 Hadamard matrices specified by the above 4 solutions, via the OD_{16} array. All 4 profiles turned out to be different and therefore these 4 matrices are inequivalent. We give below the 4-profiles of the 4 Hadamard matrices, which shows clearly that these matrices are inequivalent.

```
[ 304298064, 40886748, 70185696, 27793248, 39625110, 13071156,
15381072, 4232088, 4112892, 939120, 777588, 160440, 103278, 25368,
15792, 5040, 924, 1932, 1428, 0, 84, 12348, 84, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1680 ]

[ 104743296, 173343408, 107945712, 73945200, 32835936, 17375568,
6991152, 2837184, 1060416, 323568, 170016, 28560, 7392, 4368,
5376, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 14028 ]

[ 304220910, 40730760, 70388682, 27858516, 39489702, 13147344,
15422358, 4254348, 4081476, 935676, 779100, 162288, 102522, 21336,
15708, 4116, 1134, 420, 378, 0, 0, 12684, 42, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1680 ]

[ 104530608, 173994576, 107753856, 73236240, 33260640, 17239488,
7294224, 2763600, 1035552, 311808, 147504, 27216, 12432, 4704, 3360,
672, 0, 0, 672, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 14028 ]
```

5 Acknowledgments

This work is supported in part by a grant from the National Sciences and Engineering Research Council of Canada, NSERC. The prototype C implementation of SGA, was developed at the *Computer Algebra Research Group, CARGO, Wilfrid Laurier University*, Waterloo ON, Canada. All computations in C have been performed remotely at *SHARCnet* and WestGrid high performance computing clusters.

The Magma 2.11 computation for Hadamard matrices of order 304 has performed remotely at the *Centre de calcul formel MEDICIS, École Polytechnique* Paris, France.

Appendix A

This is available online at `http://www.cargo.wlu.ca/ODGA-AJC/`.

References

[1] R. Craigen, Hadamard Matrices and Designs, in *The CRC Handbook of Combinatorial Designs*, (eds. C. J. Colbourn and J. H. Dinitz), CRC Press, Boca Raton, Fla., 1996, pp. 370–377.

[2] K. O. Geddes, S. R. Czapor, and G. Labahn, *Algorithms for Computer Algebra*, Kluwer Academic Publishers, Boston, MA, 1992.

[3] A. V. Geramita and J. Seberry, *Orthogonal designs. Quadratic forms and Hadamard matrices*, Lecture Notes in Pure and Applied Mathematics, 45. Marcel Dekker, Inc., New York, 1979.

[4] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning* Addison-Wesley 1989

[5] M. Hall, Jr., *Combinatorial theory*, Reprint of the 1986 second edition, Wiley Classics Library, 1998. Wiley, New York.

[6] J. H. Holland, *Adaptation in natural and artificial systems, An introductory analysis with applications to biology, control, and artificial intelligence*, University of Michigan Press, Ann Arbor, Michigan, 1975.

[7] H. Kharaghani and B. Tayfeh-Rezaie, *Some new orthogonal designs in orders* 32 *and* 40, *Discrete Math.* 279 (2004), 317–324.

[8] I. S. Kotsireas and C. Koukouvinos, Orthogonal Designs via Computational Algebra, *J. Combin. Designs* (to appear).

[9] J. Seberry and R. Craigen, Orthogonal Designs, in *The CRC Handbook of Combinatorial Designs*, (eds. C. J. Colbourn and J. H. Dinitz), CRC Press, Boca Raton, Fla., 1996, pp. 400–406.

[10] J. Seberry and M. Yamada, Hadamard matrices, sequences, and block designs, in *Contemporary Design Theory: A Collection of Surveys*, eds. J. H. Dinitz and D. R. Stinson, John Wiley, New York, pp. 431-560, 1992.

(Received 10 Feb 2005)