ACS Searching for D_{4t}-Hadamard Matrices

Víctor Álvarez, José Andrés Armario, María Dolores Frau, Félix Gudiel,
Belén Güemes, Elena Martín, and Amparo Osuna

University of Sevilla, Sevilla, Spain
{valvarez,armario,mdfrau,gudiel,bguemes,emartin,aosuna}@us.es

Abstract. An Ant Colony System (ACS) looking for cocyclic Hadamard matrices over dihedral groups D_{4t} is described. The underlying weighted graph consists of the rooted trees described in [1], whose vertices are certain subsets of coboundaries. A branch of these trees defines a D_{4t}-Hadamard matrix if and only if two conditions hold: (i) $I_i = i - 1$ and, (ii) $c_i = t$, for every $2 \leq i \leq t$, where I_i and c_i denote the number of i-paths and i-intersections (see [3] for details) related to the coboundaries defining the branch. The pheromone and heuristic values of our ACS are defined in such a way that condition (i) is always satisfied, and condition (ii) is closely to be satisfied.

Keywords: Cocyclic Hadamard matrix, ant colony system, i-path, i-intersection.

1 Introduction

Hadamard matrices are square matrices with entries ±1 such that their rows are pairwise orthogonal. It is easy to prove that the size of Hadamard matrices must be 1, 2 or a multiple of 4. Nevertheless, it is an open question whether Hadamard matrices exist for every size $4t$. This is known as the *Hadamard Conjecture.* Recommended references on Hadamard matrices and their applications are [10] and more recently [12].

Actually, constructing Hadamard matrices is a difficult problem of optimization. That being so, different heuristics have been proposed to look for Hadamard matrices (see [2,6,5] for instance), but they all seem to run in exponential time $O(2^t)$.

One of the most promising techniques for solving the Hadamard Conjecture is the cocyclic approach [12], since both the search space and the time required for testing the Hadamard character of a matrix are significantly improved in this framework [12,1]. Among others, dihedral groups D_{4t} seem to provide a large amount of cocyclic Hadamard matrices (see [12] or [4], for instance). This is the reason for which we will focus on this family of groups. In the sequel, for short, cocyclic matrices over D_{4t} will be simply denoted as D_{4t}-matrices.

Experimental results in [1] suggest that one might restrict to look for D_{4t}-Hadamard matrices satisfying the *central distribution.* (this notion will be explained in detail in Section 2).

Our aim here is to use the ideas of ant colony optimization (in the sequel, ACO for brevity) in order to design an ant colony system looking for D_{4t}-Hadamard matrices satisfying the central distribution.

We organize the paper as follows. Notations and previous results are introduced in Section 2. Section 3 is devoted to the description of our ACS. Last section is devoted to examples and conclusions.

2 Describing the Rooted Trees

In what follows, we will adopt the notations and results introduced in [1], which we describe now.

Consider the dihedral group D_{4t}, given by the presentation

$$< a, b | a^{2t} = b^2 = (ab)^2 = 1 >$$

and ordering $g_1 = 1 = a^0 < \ldots < g_{2t} = a^{2t-1} < g_{2t+1} = b < \ldots < g_{4t} = a^{2t-1}b$.

A 2-cocycle over D_{4t} consists in a map $f : D_{4t} \times D_{4t} \to \{1, -1\}$ such that

$$f(g_i, g_j) f(g_i g_j, g_k) = f(g_j, g_k) f(g_i, g_j g_k), \qquad \forall\, g_i, g_j, g_k \in D_{4t}.$$

A cocyclic matrix over D_{4t} (in the sequel, D_{4t}-matrix) consists in a matrix $M_f = (f(g_i, g_j))$, f being a 2-cocycle over D_{4t}.

A basis for 2-cocycles over D_{4t} is given by $\mathcal{B} = \{\partial_a, \ldots, \partial_{a^{2t-3}b}, \beta_1, \beta_2, \gamma\}$, where ∂_g denotes the elementary coboundary related to the element g, that is

$$\partial_g(g_i, g_j) = \delta_g(g_i)\delta_g(g_j)\delta_g(g_i g_j) \qquad \text{for} \qquad \delta_g(g_i) = \begin{cases} -1, g = g_i \\ \ \ 1, g \neq g_i \end{cases}$$

$\beta_1(a^i b^l, a^j b^k) = (-1)^{ij}$, $\beta_2(a^i b^l, a^j b^k) = (-1)^{lk}$ and

$$\gamma(a^i b^l, a^j b^k) = \begin{cases} -1, l = 0 \text{ and } i + j \geq 2 \\ -1, l = 1 \text{ and } i < j \\ \ \ 1, \text{otherwise} \end{cases}$$

We will consider only D_{4t}-matrices of the type $M_f = M_{\partial_{i_1}} \ldots M_{\partial_{i_w}} \cdot R$, in terms of some coboundary matrices $M_{\partial_{i_j}}$ and the matrix $R = M_{\beta_2} M_\gamma$. There is computational evidence that most of D_{4t}-Hadamard matrices are of this type (see [9,3] for instance).

Furthermore, the cocyclic Hadamard test (which asserts that a cocyclic matrix is Hadamard if and only if the summation of each row but the first is zero, [13]) runs four times faster for this type of D_{4t}-matrices, since it suffices to check whether the summation of rows from 2 to t are zero. For clarity in the exposition, from now on, the rows whose summation is zero are simply termed *Hadamard rows*.

In [3] the Hadamard character of a cocyclic matrix is described in an equivalent way, in terms of *generalized coboundary matrices*, *i-walks* and *intersections*. We reproduce now these notions.

The *generalized coboundary matrix* $\bar{M}_{\partial_j}$ related to a elementary coboundary ∂_j consists in negating the j^{th}-row of the matrix M_{∂_j}. Note that negating a row of a matrix does not change its Hadamard character. As it is pointed out in [3], every generalized coboundary matrix $\bar{M}_{\partial_j}$ contains exactly two negative entries in each row $s \neq 1$, which are located at positions (s,i) and (s,e), for $g_e = g_s^{-1} g_i$. We will work with generalized coboundary matrices from now on.

A set $\{\bar{M}_{\partial_{i_j}} \; : \; 1 \leq j \leq w\}$ of generalized coboundary matrices defines an *i-walk* if these matrices may be ordered in a sequence $(\bar{M}_{l_1}, \ldots, \bar{M}_{l_w})$ so that consecutive matrices share exactly one negative entry at the i^{th}-row. Such a walk is called an *i-path* if the initial and final matrices do not share a common -1, and an *i-cycle* otherwise. As it is pointed out in [3], every set of generalized coboundary matrices may be uniquely partitioned into disjoint maximal i-walks. It is clear that every maximal i-path contributes two negative occurrences at the i^{th}-row.

An *i-intersection* is a position in which R and $\bar{M}_{\partial_{i_1}} \ldots \bar{M}_{\partial_{i_w}}$ share a common -1 in the i^{th}-row.

From the definitions above, a characterization of Hadamard rows (consequently, of Hadamard matrices) may be easily described in terms of the number c_i of i-paths and the number I_i of i-intersections.

Proposition 1. *[3] The i^{th} row of a D_{4t}-matrix $M = M_{\partial_{i_1}} \ldots M_{\partial_{i_w}} \cdot M_{\beta_2} \cdot M_\gamma$ is Hadamard if and only if*

$$c_i - I_i = t - i + 1, \quad 2 \leq i \leq t. \tag{1}$$

It should be desirable to know the way in which coboundaries combine to form i-paths and i-intersections. These questions have already been answered.

On one hand, as it is described in [3], for $2 \leq i \leq t$, a maximal i-walk consists of a maximal subset in $(M_{\partial_1}, \ldots, M_{\partial_{2t}})$ or $(M_{\partial_{2t+1}}, \ldots, M_{\partial_{4t}})$ formed from matrices $(\ldots, M_j, M_k, \ldots)$ such that $j \pm (i-1) \equiv k \; mod \, 2t$.

On the other hand, in [[1], Lemma 3, p.208], a complete distribution of the coboundaries in $\mathcal{B}$ which produce an intersection at a given row is described in a table, so that coboundaries which produce the same negative occurrence at a row are displayed vertically in the same column. For clarity in the reading, we note the generalized coboundary $\bar{M}_{\partial_i}$ simply by i:

row	*coboundaries*										
2					$2t$	$2t+1$					
3				$\boxed{\text{2t-1}}$	$2t$ $\boxed{2}$	$2t+1$	$\boxed{\text{2t+2}}$				
$4 \leq k \leq t$	$\boxed{\text{2t-k+2}}$	$2t-k+3$ 2	$\ldots$ $\ldots$	$2t-1$ $k-2$	$2t$ $\boxed{\text{k-1}}$	$2t+1$ $\boxed{\text{4t-k+2}}$	$2t+2$ $4t-k+1$	$\ldots$ $\ldots$	$2t+k-3$ $4t-2$	$2t+k-2$	$\boxed{\text{2t+k-1}}$

Notice that the boxed coboundary matrices do not produce any intersection at the precedent rows. Furthermore, $\bar{M}_{\partial_t}$, $\bar{M}_{\partial_{t+1}}$, $\bar{M}_{\partial_{3t}}$ and $\bar{M}_{\partial_{3t+1}}$ do not produce any intersection at all.

Though formally there are many distributions $(c_2, I_2), \ldots, (c_t, I_t)$ satisfying the Hadamard test (1), in [1] there is computational evidence that the distribution $(c_i, I_i) = (t, i-1)$ provides a large density of D_{4t}-Hadamard matrices, for $2 \leq t \leq 5$. This is termed the *central distribution*, and is expected to provide many D_{4t}-Hadamard matrices for larger values of t as well. The tables in [1] support this idea.

As it was already described in [1], the search space in the central distribution $(c_i, I_i) = (t, i-1)$, $2 \leq i \leq t$, may be represented as a forest of two rooted trees of depth $t-1$. Each level of the tree is identified to the correspondent row of the cocyclic matrix at which intersections are being counted, so that the roots of the trees are located at level 2 (corresponding to the intersections created at the second row of the cocyclic matrix).

This way the level i contains those coboundaries which must be added to the father configuration in order to get the desired $i-1$ intersections at the i^{th}-row, for $2 \leq i \leq t$.

The root of the first tree is ∂_{2t}, whereas the root of the second tree is ∂_{2t+1}, since these are the only coboundaries which may give an intersection at the second row.

As soon as one of these coboundaries is used, the other one is forbidden, since otherwise a second intersection would be introduced at the second row.

Now one must add some coboundaries to get two intersections at the third row. Notice that one and just one of $\{\partial_{2t}, \partial_{2t+1}\}$ is already used, whereas the other is forbidden.

Successively, in order to construct the nodes at level k, one must add some of the correspondent boxed coboundaries of the table, since the remaining coboundaries are either used or forbidden.

Some pictures representing these forests for $2 \leq t \leq 4$ are included in [1].

Summing up, if we assume that we are looking for D_{4t}-Hadamard matrices satisfying the central distribution, two conditions must hold:

$$I_i = i - 1 \tag{2}$$

$$c_i = t \tag{3}$$

Condition (2) means that one should find a branch in the trees above reaching the bottom level, t. But just a few of these branches define a D_{4t}-Hadamard matrix. Attending to condition (3), among all branches reaching the level t, one should select only those which inherits a subset of coboundaries such that eventually combined with some of the matrices $\bar{M}_{\partial_t}$, $\bar{M}_{\partial_{t+1}}$, $\bar{M}_{\partial_{3t}}$ and $\bar{M}_{\partial_{3t+1}}$ do produce exactly t i-walks, $2 \leq i \leq t$.

3 Defining the ACS

Sometimes solving an optimization problem is so difficult, that one restricts oneself to obtain not necessarily a global optimum but just a local one. Heuristic procedures are concerned with this purpose. A special kind of heuristics are those emulating natural behaviours, such as evolutionary algorithms (inspired by biological evolution, see [11]) and ant colony optimization (simulating the pheromone model of ants, see [8]).

Although some heuristics (in terms of image restoration [6] or even evolutionary computation [2,5]) have already been used for constructing Hadamard matrices, as far as we know ant colony optimization has not been used for this purpose yet. This is our concern here. We are going to define an ant colony system (in the sequel, ACS [7]).

To this end, we need to define a weighted graph G modeling the problem, as well as the values τ_{ij}, η_{ij}, ϕ, ρ, τ_0, q_0, α, β and the function Δ_{ij} proper of ACSs.

The underlying graph G modeling our optimization problem consists of the rooted trees T_1 and T_2 described in the Section 2.

A D_{4t}-matrix M_f satisfies the central distribution if and only if the conditions (2) and (3) are satisfied.

As it was noted before, condition (2) means that the coboundaries generating f define a branch in one of our rooted trees, ending at the bottom level, t.

Unfortunately, there is no such geometric translation for condition (3).

For instance, the rooted trees associated to the case $t = 5$ consists of 84 branches each, all of them ending at the bottom level, 5. This means that the subsets of coboundaries associated to each of these branches satisfy the condition (2). Unfortunately, only 14 of these 84 branches give raise to D_{4t}-Hadamard matrices, since only 14 of the corresponding subsets of coboundaries (eventually combined with some of $\{\partial_5, \partial_6, \partial_{15}, \partial_{16}\}$) satisfy the condition (3). These branches are listed in the table below (the subsets of added ∂_i are listed in brackets).

Branches from T_1
(10)(9)(3, 13, 18)(14)[16]
(10)(9)(3, 13, 18)(17)[15]
(10)(9)(3, 13, 18)(4, 7, 14)[5, 6, 16]
(10)(9)(3, 13, 18)(4, 7, 17)[5, 6, 15]
(10)(12)(8)(17)[5, 16]
(10)(12)(8)(4, 7, 14)[15, 16]
(10)(12)(13)(4, 7, 14)[6, 16]
(10)(12)(3, 8, 13)(4)[15]
(10)(12)(3, 8, 13)(4)[5, 6, 16]
(10)(12)(3, 8, 18)(17)[]
(10)(2, 9, 12)(8)(14)[5, 6, 15, 16]
(10)(2, 9, 12)(8)(4, 7, 17)[5, 16]
(10)(2, 9, 12)(13)(4)[16]
(10)(2, 9, 12)(13)(4)[5, 6, 15]
(10)(2, 9, 12)(18)(4, 7, 17)[5, 6]
(10)(2, 9, 12)(3, 8, 13)(14)[6, 16]

Branches from T_2
(11)(2)(3)(4, 14, 17)[6, 15]
(11)(2)(13)(7)[6, 16]
(11)(2)(13)(4, 14, 17)[5, 6]
(11)(2)(3, 13, 18)(17)[5]
(11)(2)(3, 13, 18)(17)[6, 15, 16]
(11)(2)(8, 13, 18)(7)[]
(11)(9)(8)(7, 14, 17)[5, 15]
(11)(9)(13)(4)[5, 16]
(11)(9)(13)(4, 7, 17)[5, 6]
(11)(9)(3, 13, 18)(4)[]
(11)(9)(8, 13, 18)(17)[6]
(11)(9)(8, 13, 18)(17)[5, 15, 16]
(11)(12)(3, 8, 18)(4)[6]
(11)(12)(3, 8, 18)(7)[5]
(11)(12)(3, 8, 18)(4, 14, 17)[6, 15, 16]
(11)(12)(3, 8, 18)(7, 14, 17)[5, 15, 16]

Condition (3) is not satisfied by the remainder branches. This means that given such a branch, for every possible subset of $\{\partial_5, \partial_6, \partial_{15}, \partial_{16}\}$, there exists a row i such that $c_i \neq t = 5$. For instance, the tables below show the values c_i when the set of coboundaries defining the branch $(10)(9)(3,8,13)(17)$ are combined with the subset indicated.

Added ∂_i	c_2	c_3	c_4	c_5
	4	5	5	4
[5]	5	5	5	4
[6]	5	5	4	4
[15]	5	4	6	5
[16]	4	6	5	5

Added ∂_i	c_2	c_3	c_4	c_5
$[5,6]$	5	5	4	4
$[5,15]$	6	4	6	5
$[5,16]$	5	6	5	5
$[6,15]$	6	4	5	5
$[6,16]$	5	6	4	5
$[15,16]$	4	5	6	6

Added ∂_i	c_2	c_3	c_4	c_5
$[5,6,15]$	6	4	5	5
$[5,6,16]$	5	6	4	5
$[5,15,16]$	5	5	6	6
$[6,15,16]$	5	5	5	6
$[5,6,15,16]$	5	5	5	6

Now we are in conditions to define the values τ_{ij} and η_{ij}.

The heuristic value η_{ij} is defined attending to condition (3), so that $\eta_{ij} = \frac{1}{\sum\limits_{k=2}^{l} |t - c_k|}$, where l indicates the level of vertex v_j. This way, the nearer the path is to the central distribution $(t, \ldots, t)$, the higher η_{ij} is.

Initially, $\tau_{ij} = \tau_0 = 0.25$. We define $\Delta_{ij} = \tau_{ij} + 3\frac{\phi}{\rho}|\tau_{ij} - \tau_0|$, so that the pheromone values are updated for the set of edges belonging to the largest path among the ants' traversals by means of the formula $\tau_{ij} = (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij} = \tau_{ij} + 3\phi|\tau_{ij} - \tau_0|$. Thus, although the local pheromone update has taken place before, the final value of τ_{ij} is greater or equal than its initial value, excepting the value of the last edge of the path, which is settled to 0. Consequently, this edge will never be used again. This is not a source of difficulties for the algorithm, since our graph consists of trees, and hence there is no edge going further.

We set $\alpha = 0.75$ and $\beta = 0.25$, so that the relative importance of pheromone versus heuristic information is settled accordingly to our purposes, since we need to get a branch reaching the bottom level t.

The evaporation rate is $\rho = 0.5$, the pheromone decay coefficient is $\phi = 0.5$, the probability of choosing the "best" edge is $q_0 = 0.75$. These values have been fixed experimentally.

Every iteration, the ants looks for good paths. For paths reaching at least level $t-1$, a local search is performed. This procedure consists in an exhaustive search from the vertex of the branch located at level $t-1$, combining with the 16 subsets of $\{\partial_t, \partial_{t+1}, \partial_{3t}, \partial_{3t+1}\}$. No matter the result of the search is, the weight of the edge arriving to the level $t-1$ is settled to 0, since now there is no chance to get not traversed paths from it. Whenever a D_{4t}-Hadamard matrix is found while performing the local search, it is added to a list *hadmat*.

The algorithm stops as soon as a the list *hadmat* is not empty.

We include now a pseudo-code for our ACS.

Algorithm 1. *ACS searching for D_{4t}-Hadamard matrices.*

```
Input: an integer $t$
Output: a list $hadmat$ of $D_{4t}$-Hadamard matrices.

$hadmat \leftarrow \emptyset$
while $hadmat$ is empty {
    for $i$ from 1 to $m$ do {
        $trav_i \leftarrow$ traversal of ant $i$
        if $trav_i$ reaches level $t-1$ then local_search($trav_i$)
        set to 0 the weight of the last edge of $trav_i$
    }
    actualize the weight of the edges of the best traversal
}
$hadmat$
```

In the following section we include some executions and final comments.

4 Examples

All the calculations of this section have been worked out in Mathematica 4.0, running on a *Pentium IV 2.400 Mhz DIMM DDR266 512 MB.*

We have fixed $m = 5$, so that every iteration 5 traversals are performed.

For every $3 \leq t \leq 8$, we have performed 10 trials looking for D_{4t}-Hadamard matrices. The table below shows the fewest, largest and average number of iterations required, the best, worst and average time required and the minimum, maximum and average number of D_{4t}-Hadamard matrices found in these calculations.

t	Fewest	Largest	Av.Iter.	Best	Worst	Av.Time	min	max	Av.
3	1	1	1	$0.02''$	$0.11''$	$0.08''$	10	10	10
4	1	1	1	$0.48''$	$0.66''$	$0.59''$	14	22	19.6
5	1	1	1	$1.56''$	$2.34''$	$1.86''$	2	4	3
6	1	1	1	$1.01''$	$7.43''$	$3.59''$	1	4	2.5
7	2	14	6.1	$21.013''$	$3'18''$	$1'20''$	1	2	1.1
8	1	2	1.2	$1.93''$	$36.31''$	$15.41''$	1	2	1.1

For values $t \geq 9$, the algorithm does not find a D_{4t}-Hadamard matrix in reasonable time (more than 100 iterations are required). Nevertheless, the traversals reach the bottom level t most of times. Unfortunately, the subsets of coboundaries do not produce the required number of i-paths, for some row i. The table below shows how many of 500 traversals for $t = 9$ have ended at the corresponding level, as well as the amount of traversals for which the difference $\sum\limits_{i=2}^{t} |t - c_i|$ is the indicated.

Level			$\sum\limits_{i=2}^{t} \lvert t-c_i \rvert$											
6	7	8	14	12	11	10	9	8	7	6	5	4	3	2	1
10	15	475	1	5	9	14	39	106	122	61	52	66	17	7	1

Consequently, most of the traversals (475 from 500) satisfy the condition (2). This means that the definition of the pheromone values τ_{ij} is fine.

On the other hand, the condition (3) is never satisfied. This fact suggests that the formula for the heuristic values η_{ij} should be redefined somehow. We will deal with this problem in a near future.

Acknowledgments. All authors are partially supported by the research projects FQM–296 and P07-FQM-02980 from Junta de Andalucía and MTM2008-06578 from Ministerio de Ciencia e Innovación (Spain).

References

1. Álvarez, V., Armario, J.A., Frau, M.D., Gudiel, F., Osuna, A.: Rooted trees searching for cocyclic Hadamard matrices over D_{4t}. In: Bras-Amorós, M., Høholdt, T. (eds.) AAECC-18. LNCS, vol. 5527, pp. 204–214. Springer, Heidelberg (2009)
2. Álvarez, V., Armario, J.A., Frau, M.D., Real, P.: A genetic algorithm for cocyclic Hadamard matrices. In: Fossorier, M.P.C., Imai, H., Lin, S., Poli, A. (eds.) AAECC 16. LNCS, vol. 3857, pp. 144–153. Springer, Heidelberg (2006)
3. Álvarez, V., Armario, J.A., Frau, M.D., Real, P.: A system of equations for describing cocyclic Hadamard matrices. J. of Comb. Des. 16(4), 276–290 (2008)
4. Álvarez, V., Armario, J.A., Frau, M.D., Real, P.: The homological reduction method for computing cocyclic Hadamard matrices. J. Symb. Comput. 44, 558–570 (2009)
5. Álvarez, V., Frau, M.D., Osuna, A.: A genetic algorithm with guided reproduction for constructing cocyclic Hadamard matrices. In: ICANNGA 2009. LNCS, vol. 5495, pp. 150–160. Springer, Heidelberg (2009)
6. Baliga, A., Chua, J.: Self-dual codes using image resoration techniques. In: Bozta, S., Sphparlinski, I.E. (eds.) AAECC 14. LNCS, vol. 2227, pp. 46–56. Springer, Heidelberg (2001)
7. Dorigo, M., Gambardella, L.M.: Ant Colony System: A cooperative learning approach to the traveling salesman problem. IEEE Trans. on Evol. Comp. 1(1), 53–66 (1997)
8. Dorigo, M., Stützle, T.: Ant colony optimization. The MIT Press, Cambridge (2004)
9. Flannery, D.L.: Cocyclic Hadamard matrices and Hadamard groups are equivalent. J. Algebra 192, 749–779 (1997)
10. Hedayat, A., Wallis, W.D.: Hadamard Matrices and Their Applications. Ann. Stat. 6, 1184–1238 (1978)
11. Holland, J.H.: Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor (1975)
12. Horadam, K.J.: Hadamard matrices and their applications. Princeton University Press, Princeton (2006)
13. Horadam, K.J., de Launey, W.: Generation of cocyclic Hadamard matrices. In: Computational algebra and number theory (Sydney, 1992). Math. Appl, vol. 325, pp. 279–290. Kluwer Acad. Publ., Dordrecht (1995)