



**HAL**  
open science

## Meta-Heuristic Search and Square Erickson Matrices

Denis Robilliard, Amine Boumaza, Virginie Marion-Poty

► **To cite this version:**

Denis Robilliard, Amine Boumaza, Virginie Marion-Poty. Meta-Heuristic Search and Square Erickson Matrices. 2009. hal-00448392v2

**HAL Id: hal-00448392**

**<https://hal.science/hal-00448392v2>**

Preprint submitted on 20 Jan 2010 (v2), last revised 4 May 2010 (v3)

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Meta-Heuristic Search and Square Erickson Matrices

Denis Robilliard and Amine Boumaza and Virginie Marion-Poty

**Abstract**—A Ramsey theory problem, that can be seen as a 2 dimensional extension of the Van der Waerden theorem, was posed by Martin J. Erickson in his book [1]: “find the minimum  $n$  such that if the lattice points of  $[n] \times [n]$  are two-colored, there exist four points of one color lying on the vertices of a square with sides parallel to the axes”. This was solved recently by Bacher and Eliahou in 2009 [2], who showed that  $n = 15$ . In this paper we tackle a derived version of this problem, searching for the minimum  $n$  that forces the existence of a monochromatic  $[3] \times [3]$  subgrid of  $[n] \times [n]$  of the form  $\{i, i + t, i + 2t\} \times \{j, j + t, j + 2t\}$ . We use meta-heuristics on this open problem to find instances of 2-colorations without monochromatic  $[3] \times [3]$  subgrid of the above form, setting a lower bound on  $n$ . In particular we found such a binary square grid of size 662, implying that  $n > 662$ .

## I. INTRODUCTION

Ramsey theory is a branch of mathematics that studies the existence of orderly substructures in large chaotic structures. In fact, the existence of orderly sub-structures is guaranteed by the theory: “complete disorder is impossible<sup>1</sup>”. The central question Ramsey theorists struggle to answer is: how large must be the structure such that it contains orderly substructures?

For example, consider a sequence of symbols, colored randomly using a fixed number of colors. Although this sequence may be generated as randomly as possible, if the sequence is sufficiently long there must exist a sub-sequence of a given length that is in arithmetic progression whose symbols are all colored the same.

More formally Van der Waerden’s theorem states that considering  $k > 1$  symbols (or colours), there exist an integer  $N$  such that any  $k$ -coloured sequence of length at least  $N$  contains a sub-sequence of length  $l > 1$  in arithmetic progression (at indices  $i, i + t, \dots, i + (l - 1)t$ ) that is monochromatic (i.e. contains only one repeated colour). The smallest  $N$  for which this holds is called the Van der Waerden number  $W(k, l)$ . Such numbers are known only for few values of  $k$  and  $l$  [3]. The 6<sup>th</sup>, and, largest known, Van der Waerden number for 2 colors was found in 2008 by Kouril et al. [4] and is  $W(2, 6) = 1132$ .

Tibor Gallai [5] extended the above theorem to the  $d$ -dimensional case. Within this framework, M. J. Erickson[1] posed the following problem:

*Open problem 1:* “Find the minimum  $n$  such that if the  $n^2$  lattice points of  $[n] \times [n]$  are two-colored, there exist four points of one color lying on the vertices of a square with sides parallel to the coordinate axes.”

Denis Robilliard, Amine Boumaza and Virginie Marion-Poty are with the Univ Lille Nord de France, Laboratoire d’Informatique du Littoral, Maison de la Recherche Blaise Pascal, 50 rue Ferdinand Buisson - BP 7119, 62228 CALAIS Cedex, France ; email: {robillia,boumaza,poty}@lil.univ-littoral.fr

<sup>1</sup>Quote attributed to Theodore Motzkin

This problem can be slightly rephrased as the following: “find the smallest  $n$  such that there are no monochromatic 2-squares in a binary matrix of size  $n \times n$ ”. A monochromatic 2-square is a sub-matrix  $S$  with row indices  $\{i, i + t\}$  and column indexes  $\{j, j + t\}$  with  $t \geq 1$  whose 4 corners are colored the same. Following [2] we pose the following definition:

*Definition 1:* An Erickson matrix is a binary matrix containing no monochromatic (or constant) 2-squares.

In the following, we will use the term matrices to refer to binary square matrices.

In [2] the authors give the exact value of  $n = 15$  for the above problem. This value was determined by computational means and a clever exploration of the search space, represented as a binary tree and using a special data structure that prevents from storing the tree nodes. Furthermore the authors list all Erickson matrices for  $n = 14$ .

Similarly to the Van der Waerden numbers, one can search for higher order Erickson matrices i.e. that are free of constant  $s$ -squares, where an  $s$ -square in  $[n] \times [n]$  is any square subgrid of the form  $\{i, i + l, \dots, i + (s - 1)l\} \times \{j, j + l, \dots, j + (s - 1)l\}$  with  $s^2$  points.

*Definition 2:* For any positive integer  $k > 1$ , let us denote by  $n(k, s)$  the smallest integer  $n$  for which any  $k$ -coloring of the grid  $[n] \times [n]$  contains a monochromatic  $s$ -square.

It follows from a theorem of Gallai [5] that  $n(k, s)$  is finite. The actual order of magnitude of  $n(k, s)$  is completely unknown at the time of this writing. For instance, before the result  $n(2, 2) = 15$  of [2], it was only known that:  $13 \leq n(2, 2) \leq \min(W(2, 8), 5 \cdot 2^{2^{40}})$  (where  $W(2, 8)$  is a still unknown Van der Waerden number).

Following their discovery of  $n(2, 2) = 15$ , Bacher and Eliahou proposed as open problems the cases  $n(3, 2)$  and  $n(2, 3)$ . In the present work, our goal is to find a lower bound for  $n(2, 3)$ . More precisely, we aim to provide a lower bound  $n(2, 3) \geq n_0 + 1$  by constructing a specific matrix of size  $n_0$  which is free of monochromatic 3-squares of the form  $\{i, i + l, i + 2l\} \times \{j, j + l, j + 2l\}$ . We first focus our interest on the closed 2-squares problem for which we have the exact bound, in order to find and validate different search methods.

The algorithm proposed by Bacher and Eliahou in [2] explores the search space exhaustively starting from a  $2 \times 2$  matrix constructing successively larger matrices by adding a so-called “elbow” (a column at the right and a row on the bottom). In [4] Kouril and Paul used a dedicated SAT solver to find  $W(2, 6)$ , a method which could also be used for Erickson matrices, as we show in Section II-A. However these methods reach their limits when dealing with large matrices. That is why we choose to tackle this problem using heuristic methods.

We study the behaviour of several meta-heuristics for different sizes of matrices. In doing so, we observe that for sizes 13 and 14, the problem of finding Erickson matrices could be of some interest for the evolutionary and meta-heuristics communities as a benchmark problem. Even though the problem is easy for a SAT solver, we found that for these sizes the problem can be difficult for different meta-heuristics within a limited number of evaluations.

## II. SEARCHING FOR ERICKSON MATRICES

As our goal is to find lower bounds, we consider a question derived from problem 1, defined as follows:

*Open problem 2:* “Let  $n < 15$  an integer, find a square Erickson matrix of size  $n$ .”

This provides us with a range of problems of increasing difficulty, depending on  $n$ . From [2] we know the number of solutions<sup>2</sup> to problem 2, denoted  $Er(n)$ :

| $n$       | $Er(n)$      |
|-----------|--------------|
| 2         | 7            |
| 3         | 138          |
| 4         | 5490         |
| 5         | 390856       |
| 6         | 29169574     |
| 7         | 1533415720   |
| 8         | 29085496072  |
| 9         | 156515895928 |
| 10        | 54978562276  |
| 11        | 2510360996   |
| 12        | 1990028      |
| 13        | 570132       |
| 14        | 116114       |
| $\geq 15$ | 0            |

For the smallest values of  $n$ , the problem is overly easy due to the small size of the search space, but the density of optimal solutions decreases when  $n$  gets closer to 15. For instance, when  $n = 14$ , which is the largest size for which there exist Erickson matrices, the number of solutions is slightly less than  $2^{18}$  while the search space is of dimension  $2^{196}$ .

### A. Exact resolution with a SAT solver

Problem 2 can be easily expressed as a boolean satisfiability problem in conjunctive normal form, where the variables are the coefficients of the matrix. For each possible 2-square, we define two clauses that forbid the 2-square to be monochromatic:

$$(x_{i,j} \vee x_{i+t,j} \vee x_{i,j+t} \vee x_{i+t,j+t})$$

$$(\neg x_{i,j} \vee \neg x_{i+t,j} \vee \neg x_{i,j+t} \vee \neg x_{i+t,j+t})$$

where  $x_{i,j}$  denotes the matrix coefficient at indices  $(i, j)$ , and  $t$  is the size of the 2-square.

<sup>2</sup>This is indeed the number of matrices with upper left coefficient equal to 0: the exact number of solutions is doubled.

We used the `march_pl` SAT solver developed at Delft University<sup>3</sup>. For the hardest instance, where the matrix size equals 14, the problem was expressed as 1638 clauses on 196 variables, and a solution was found in about 2.5 minutes on a standard PC (this obviously yields solutions for smaller instances). Furthermore we verified the non existence of monochromatic 2-squares on matrices of size 15, where the problem was expressed as 2030 clauses on 225 variables, which could be done in roughly 20 minutes.

However this method cannot be applied on matrices as large as those considered in Section III. For example, if we express the problem of finding a  $300 \times 300$  matrix free of monochromatic 3-squares, we obtain 8732592 clauses on 90000 variables, which is untractable with standard SAT solvers. This is why we investigate the performance of heuristic search methods whose behaviour is less dependent on the size of the problem.

### B. Heuristics search methods

The different meta-heuristics we used to solve problem 2 are: a local search hybridized with path relinking, a simulated annealing algorithm, and a simulated annealing hybridized with local search. All these heuristics update a search point (a matrix) using a neighborhood operator. Depending on the fitness of the newly generated point and the algorithm used, the new point replaces or not the original one. The exact algorithms will be presented shortly, in the meantime we discuss the features that are common to all of them.

#### 1) Common features:

a) *Initialization method:* Let us consider the range of possible square binary matrices of any given size, ordered by increasing number of 1 coefficients. Clearly Erickson matrices cannot be located far from the middle of this range: at the extremes, a matrix contains almost only 0's or 1's and thus contains many constant 2-squares. In other words the ratio of 0 coefficients over 1 coefficients in an Erickson matrix cannot be far from 1, i.e. Erickson matrices are almost balanced in 0 and 1. As an example, in size 14 the proportion of balanced matrices is given by the ratio of the binomial coefficient  $(\binom{14^2}{2})/2$  over the number of matrices  $2^{14^2}$ , which represents 5.7% of the search space but happens to include about 20% of Erickson matrices [6].

Thus, in order to speedup the search, we chose random balanced matrices as initial solutions for all our algorithms.

b) *Fitness function and neighborhood operator:* The fitness function returns the number of monochromatic 2-squares present in the matrix, thus we work in a minimization context. Our basic neighborhood operator is the simple bit flip. When a bit flip is performed we only compute the fitness adjustment based on the 2-squares to which the flipped bit belongs. This is more efficient than recomputing the fitness over the whole matrix.

#### 2) Algorithms details:

<sup>3</sup><http://www.st.ewi.tudelft.nl/sat>

a) *Local search and the GRASP framework*: Our local search operator is defined in the following way: if a single bit flip improves the fitness then it is performed, otherwise we search for a second bit-flip such that the combined two flips improve the fitness. Only the bits belonging to monochromatic 2-squares of the current search point (matrix) are considered, since flipping other bits can only degrade the fitness.

On this local search we apply a path relinking heuristic similar to the GRASP method proposed by [7]. We perform the local search on an initial solution, then we explore the path linking this solution to one picked randomly in an elite archive. At each step of the linking path we choose to flip the bit that leads to the best point. We tested four archive sizes: 5, 25, 50 and 100. A new elite solution replaces the closest one (with respect to the Hamming distance) with a worse fitness, if any in the archive.

b) *Simulated Annealing*: Our simulated annealing algorithm is quite standard (see [8]). We use a static cooling schedule, defined by the following temperature formula:

$$t_k = A / \log(1 + (k/B)^C)$$

where  $k$  is the iteration counter and  $A, B, C$  are constants. This formula provides the initial temperature for  $k = 1$ . The values for  $A, B$  and  $C$  are given further in Section II-C for the 2-squares problem and in Section III for the 3-squares problem. After each decrease of the temperature, we perform  $4 \times n^2$  (where  $n$  is the size of the matrix) iterations at constant temperature. We remind that the probability of acceptance of a worse search point  $x'$  when current point is  $x$  is given by:

$$Pr(x \leftarrow x') = \exp\left(\frac{fitness(x) - fitness(x')}{t_k}\right)$$

We test three variants of the perturbation operator:

- in the first one, a new search point is generated by randomly flipping one bit of the matrix
- in the second one, we explore only the set of balanced matrices, thus we flip randomly two bits maintaining the matrix balanced
- in the last one, we randomly flip two bits of the matrix.

c) *Large Step Markov Chain (or LSMC)*: Our third algorithm is inspired from [9]. It can be viewed as a variant of simulated annealing, where a new search point is constructed by:

- first, applying a medium size partly random perturbation (named “kick” in the original article from Martin et al.) to the current point; at this stage the fitness of the new point is probably worse than its predecessor;
- then, optimizing the resulting point with a local search heuristic.

The final point is accepted or rejected as in the simulated annealing method. Thus the main difference with simulated annealing is that we focus on local optima of the local search heuristic. This general framework is also known as *memetic* algorithms [10].

In our case, the cooling schedule was the same as in the previous method. We tested two variants: one where the “kick” perturbation was the inversion of 3 random bits belonging to monochromatic 2-squares; and the other where the 3 inverted bits were chosen randomly in the whole matrix. Such “kick” perturbations unfortunately do not guarantee that the loss of fitness remains limited, because of the high level of dependency between coefficients of the matrix. Thus we are not in a context as favourable as in the case of the euclidean TSP studied in [9].

### C. Experiments

In this section we compare the performance of the different search methods. Experiments were conducted with the following setting:

- matrices of size  $10 \leq n \leq 14$ ,
- maximum number of evaluations  $5 \cdot 10^7$ ,
- and 30 independent runs.

The simulated annealing and LSMC algorithms used parameters  $A = 1.5$ ,  $B = 1 \cdot 10^5$  and  $C = 1$ .

The results from the simulated annealing algorithm are in Table I, those from LSMC are in Table II and those from the GRASP method are given in Table III. These tables are structured as follows:

- column “size” is the matrix size
- the second column indicates the variant of the heuristic
- column “best fit.” is the minimum fitness reached during the 30 test runs
- “# optima” is the number of runs that reached the “best fit.” value
- “opt. cost” is the mean number of evaluations used by the “# optima” runs with standard deviation given in the same column between parentheses (values are to be multiplied by  $10^4$ )
- “final fitness” is the final fitness averaged on the 30 runs with standard deviation given in the same column between parentheses.

From these experiments we can observe that:

- Sizes 10 and 11 are fairly simple instances, all algorithms found optimal solutions at each run, however with different evaluation cost, LSMC dominating the others.
- For size 12 an optimal solution is no more found at each run, except by the simulated annealing. However some runs of the other algorithms were able to find an optimal solution, GRASP being the weakest.
- For sizes 13 and 14, only the simulated annealing was able to find optimal solutions (such an optimal size 14 matrix is illustrated in Figure 1).
- The best LSMC variant uses the “kick” perturbation defined randomly on the whole matrix.
- The best simulated annealing variant uses the 1 bit perturbation. It is also to be noted that using a balanced 2 bits perturbation is superior to a random 2 bits change, as it was expected due to the greater density

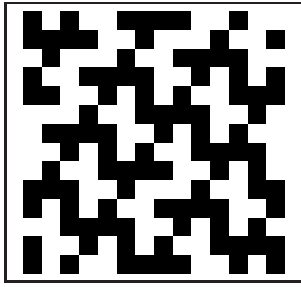


Fig. 1. An Erickson matrix of size 14 found by simulated annealing.

of optimal solutions in balanced matrices. However 1 bit perturbation is still superior.

- Overall, simulated annealing performed the best, although it is more costly in small instances (sizes 10 and 11). GRASP was the weakest heuristics, and the archive size did not strongly influence its success rate.

Figures 2–5 illustrate the behaviour of the best variant on different problem sizes of each heuristic: GRASP with an archive size of 100, Simulated Annealing (SA) with one bit perturbation and LSMC with random kick. The curves represent average fitness values over 30 runs. We notice that even though GRASP and LSMC start at lower fitness values than SA, their fitness decrease is slower than SA on sizes 12 and up. As it is shown in these figures, GRASP appears to be the weakest heuristic on these problems.

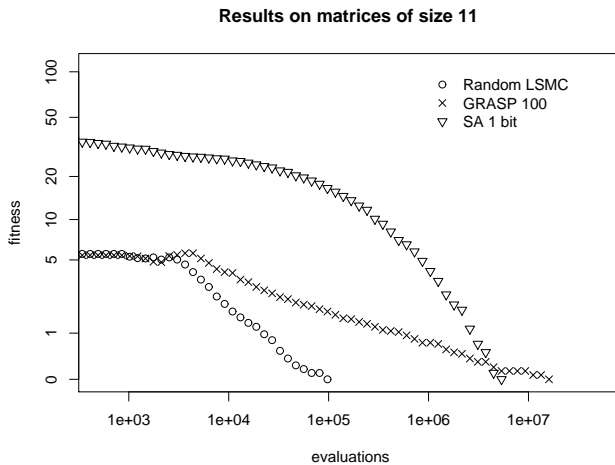


Fig. 2. Evolution of the mean fitness for the best variant of the three heuristics for size 11. Both axes are in log scale.

### III. A LOWER BOUND FOR $n(2, 3)$

We can now tackle the search for a lower bound to  $n(2, 3)$ . We remind that we are now searching for square binary matrices, as large as possible, that do not contain any constant 3-squares of the form  $\{i, i + l, i + 2l\} \times \{j, j + l, j + 2l\}$ . First we consider some comparison elements about the fitness landscapes related to  $n(2, 2)$  and  $n(2, 3)$ , then we report the results of our experiments.

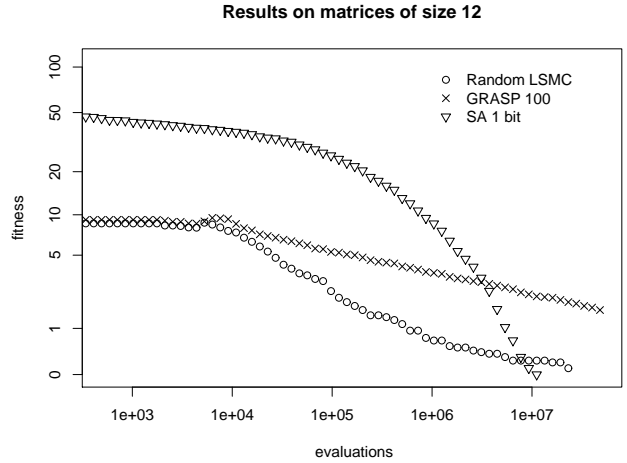


Fig. 3. Evolution of the mean fitness for the best variant of the three heuristics for size 12. Both axes are in log scale.

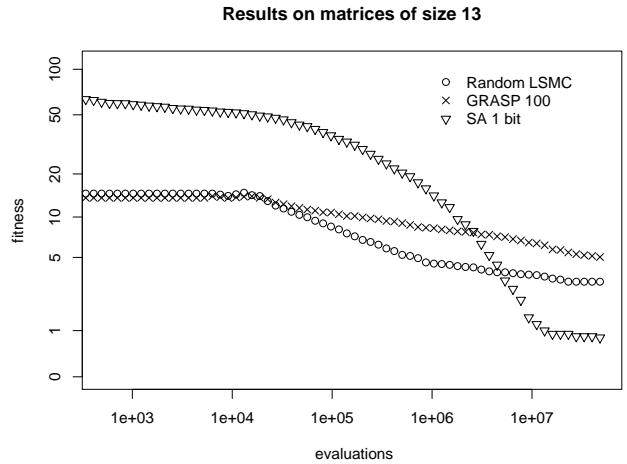


Fig. 4. Evolution of the mean fitness for the best variant of the three heuristics for size 13. Both axes are in log scale.

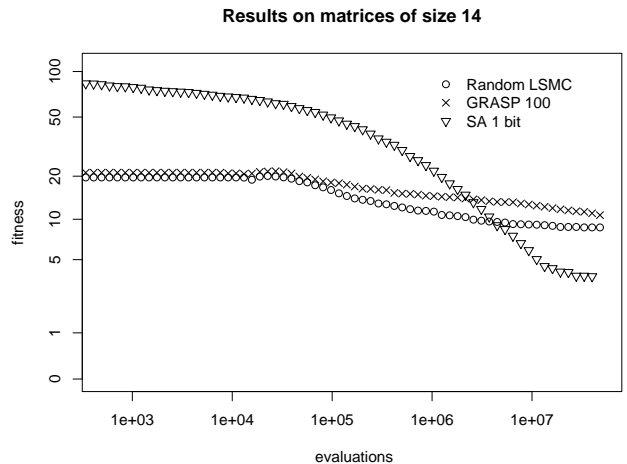


Fig. 5. Evolution of the mean fitness for the best variant of the three heuristics for size 14. Both axes are in log scale.



TABLE I

SIMULATED ANNEALING RESULTS FOR DIFFERENT PERTURBATION OPERATORS ("1" FOR 1 BIT MUTATION, "2\*" FOR BALANCED 2 BITS MUTATION AND "2" FOR RANDOM 2 BITS MUTATION).

| size | variant | best | # opt. | opt. cost $\times 10^4$ |        | final fit. |        |
|------|---------|------|--------|-------------------------|--------|------------|--------|
| 10   | 1       | 0    | 30     | 175                     | (30)   | 0          | (0)    |
|      | 2*      | 0    | 30     | 187                     | (52)   | 0          | (0)    |
|      | 2       | 0    | 30     | 197                     | (45)   | 0          | (0)    |
| 11   | 1       | 0    | 30     | 363                     | (70)   | 0          | (0)    |
|      | 2*      | 0    | 30     | 411                     | (91)   | 0          | (0)    |
|      | 2       | 0    | 30     | 420                     | (93)   | 0          | (0)    |
| 12   | 1       | 0    | 30     | 707                     | (150)  | 0          | (0)    |
|      | 2*      | 0    | 30     | 1020                    | (220)  | 0          | (0)    |
|      | 2       | 0    | 30     | 1070                    | (310)  | 0          | (0)    |
| 13   | 1       | 0    | 6      | 2060                    | (1500) | 0.80       | (0.41) |
|      | 2*      | 0    | 6      | 1350                    | (810)  | 0.8        | (0.41) |
|      | 2       | 0    | 3      | 1220                    | (740)  | 0.93       | (0.37) |
| 14   | 1       | 0    | 3      | 2100                    | (850)  | 3.67       | (2.20) |
|      | 2*      | 0    | 2      | 928                     | (240)  | 4.83       | (1.6)  |
|      | 2       | 1    | 1      | 1640                    | (NA)   | 5.27       | (0.94) |

TABLE II

LARGE STEP MARKOV CHAIN RESULTS FOR 2 VARIANTS ("2-SQUARE" FOR A KICK MUTATION ONLY ON BITS BELONGING TO CONSTANT 2-SQUARES, "RANDOM" FOR A KICK MUTATION ON RANDOM BITS).

| size | variant  | best | # opt. | opt. cost $\times 10^4$ |        | final fit. |        |
|------|----------|------|--------|-------------------------|--------|------------|--------|
| 10   | 2-square | 0    | 30     | 0.31                    | (0.21) | 0          | (0)    |
|      | random   | 0    | 30     | 0.42                    | (0.29) | 0          | (0)    |
| 11   | 2-square | 0    | 30     | 3.31                    | (2.4)  | 0          | (0)    |
|      | random   | 0    | 30     | 3.26                    | (2.7)  | 0          | (0)    |
| 12   | 2-square | 0    | 18     | 856                     | (1300) | 0.433      | (0.57) |
|      | random   | 0    | 27     | 419                     | (680)  | 0.1        | (0.31) |
| 13   | 2-square | 1    | 1      | 4040                    | (NA)   | 3.5        | (1.1)  |
|      | random   | 1    | 1      | 1730                    | (NA)   | 3.1        | (0.84) |
| 14   | 2-square | 7    | 2      | 753                     | (720)  | 9.27       | (1.2)  |
|      | random   | 6    | 2      | 1550                    | (1900) | 8.57       | (1.6)  |

TABLE III

GRASP RESULTS FOR DIFFERENT ARCHIVE SIZES.

| size | archive | best | # optima | opt. cost $\times 10^4$ |        | final fit. |        |
|------|---------|------|----------|-------------------------|--------|------------|--------|
| 10   | 5       | 0    | 30       | 3.31                    | (3.8)  | 0          | (0)    |
|      | 25      | 0    | 30       | 3.84                    | (4.7)  | 0          | (0)    |
|      | 50      | 0    | 30       | 3.84                    | (4.7)  | 0          | (0)    |
|      | 100     | 0    | 30       | 3.84                    | (4.7)  | 0          | (0)    |
| 11   | 5       | 0    | 30       | 384                     | (390)  | 0          | (0)    |
|      | 25      | 0    | 30       | 254                     | (350)  | 0          | (0)    |
|      | 50      | 0    | 30       | 242                     | (230)  | 0          | (0)    |
|      | 100     | 0    | 30       | 329                     | (390)  | 0          | (0)    |
| 12   | 5       | 1    | 14       | 2740                    | (1400) | 1.57       | (0.57) |
|      | 25      | 0    | 2        | 2690                    | (400)  | 1.57       | (0.73) |
|      | 50      | 0    | 2        | 3200                    | (610)  | 1.7        | (0.75) |
|      | 100     | 0    | 2        | 3410                    | (310)  | 1.63       | (0.72) |
| 13   | 5       | 3    | 1        | 3840                    | (NA)   | 5          | (0.9)  |
|      | 25      | 3    | 2        | 775                     | (250)  | 5          | (0.1)  |
|      | 50      | 3    | 2        | 2530                    | (2000) | 5.13       | (0.97) |
|      | 100     | 2    | 1        | 4320                    | (NA)   | 5.03       | (1)    |
| 14   | 5       | 8    | 2        | 2570                    | (2900) | 10.6       | (1.1)  |
|      | 25      | 8    | 2        | 3670                    | (840)  | 10.7       | (1.2)  |
|      | 50      | 8    | 4        | 2240                    | (1100) | 10.4       | (1.3)  |
|      | 100     | 8    | 2        | 3240                    | (1400) | 10.6       | (1.1)  |

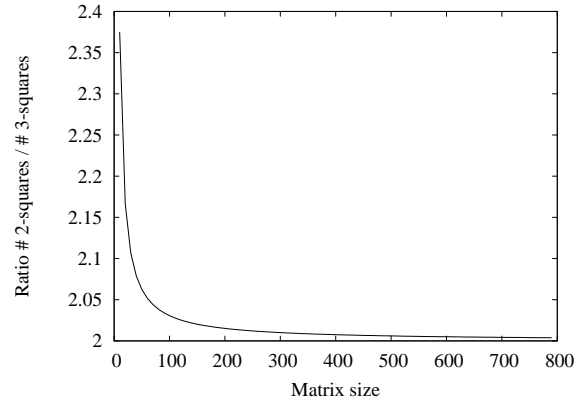


Fig. 6. The ratio of possible 2-squares over 3-squares plotted in function of the matrix size.

d) *Fitness landscape*: At first we can observe that a bit in the 4 corners of the matrix obviously belongs to fewer 2-squares (respectively 3-squares) than a bit located closer to the center of the matrix. In order to simplify our reasoning, we propose to consider the average number of 2-squares and 3-squares to which a bit belongs. This can be computed easily, and we observe that on average the number of 2-squares tends to be twice the number of 3-squares when the matrix size grows, as illustrated in Figure 6.

In terms of fitness landscape, this suggests that the interdependence between bits (also called *epistasis*) is lower when trying to get rid of monochromatic 3-squares than when working on 2-squares, and thus the search for optimal matrices of a given size is easier when working on  $n(2, 3)$  than on  $n(2, 2)$ . However the overall problem difficulty increases due to the large size of the matrices involved in searching an as large as possible bound for  $n(2, 3)$ .

e) *Experiments*: From the experiments on constant 2-squares reported in the previous section, we selected the simulated annealing with single bit perturbation as the most promising heuristic to search for a lower bound for  $n(2, 3)$ .

We had to adapt the cooling schedule to this new problem, as reaching low fitness values for large matrices required staying longer at a lower temperature than in the 2-square instances. The temperature parameters are now:  $A = 1$ ,  $B = 1.001$  and  $C = 0.7$ , and the stopping criterion has also been changed such that the algorithm now ends when there is no fitness improvement for  $2 \cdot 10^8$  evaluations.

We explored matrices of size ranging from 30 and up. The largest matrix avoiding monochromatic 3-squares we found is of size 662 and was obtained in 4 hours on an Itanium-2 processor running at 1.5GHz. It is illustrated in Figure 7, interested readers may download it at <http://lil.univ-littoral.fr/~robillia/matrices/662>.

#### IV. CONCLUSION

In this paper we focused on the problem of finding a lower bound for  $n(2, 3)$  using heuristic methods. First we worked on the recently closed problem  $n(2, 2)$ , testing different algorithms for searching for Erickson matrices. We found

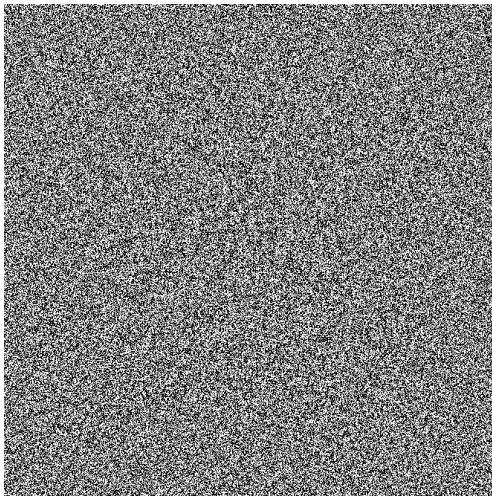


Fig. 7. A size 662 matrix free from monochromatic 3-squares.

that simulated annealing was the most promising, as it was the only one which reached optimal solutions on sizes 13 and 14. A standard SAT solver was also tested and found to be very efficient on the  $n(2, 2)$  problem but could not process matrices as large as those involved for  $n(2, 3)$ .

We gave some insights, based on epistasis, on the relative difficulty of these problems. From the preliminary results we chose to apply simulated annealing to our initial problem, and we found a lower bound:  $n(2, 3) > 662$ . To our knowledge, this work is the first attempt to provide such a lower bound.

We did not use population based evolutionary algorithms per se, since we did not find a satisfying crossover operator due to the inter-dependence of genes (matrix coefficients). However this could constitute an interesting research question.

From this study we think that searching for Erickson matrices could be an interesting benchmark for the evolutionary and meta-heuristics communities. Furthermore, heuristics seem to be powerful tools to tackle similar problems originating from Ramsey theory.

#### ACKNOWLEDGMENTS

We are grateful to S. Eliahou for introducing us to the fascinating intricacies of Erickson matrices.

#### REFERENCES

- [1] M. Erickson, *Introduction to Combinatorics*, ser. Series in Discrete Mathematics and Optimization. Wiley-Interscience, 1996.
- [2] R. Bacher and S. Eliahou, "Extremal binary matrices without constant 2-squares," Univ Lille Nord de France, Tech. Rep. "Cahiers du LMPA Joseph Liouville" N° 404, 2009, submitted.
- [3] P. Herwig, M. Heule, P. V. Lambalgen, and H. V. Maaren, "A new method to construct lower bounds for van der Waerden numbers," *The Electronic Journal of Combinatorics*, vol. 14, no. 1, p. #R6, 2007.
- [4] M. Kouril and J. L. Paul, "The van der Waerden number  $W(2, 6)$  is 1132," *Experimental Mathematics*, vol. 17, no. 1, pp. 53–61, 2008.
- [5] P. Anderson, "A generalization of Baudet's conjecture (van der Waerden's theorem)," *Amer. Math. Monthly*, no. 83, pp. 359–361, 1976.
- [6] S. Eliahou, 2009, personal communication.
- [7] M. Laguna and R. Martí, "Grasp and path relinking for 2-layer straight line crossing minimization," *INFORMS Journal on Computing*, vol. 11, pp. 44–52, 1999.

- [8] E. Aarts and J. K. Lenstra, Eds., *Local Search in Combinatorial Optimization*. New York, NY, USA: John Wiley & Sons, Inc., 1997.
- [9] O. Martin, S. W. Otto, and E. W. Felten, "Large-step markov chains for the traveling salesman problem," *Complex Systems*, vol. 5, pp. 299–326, 1991.
- [10] N. J. Radcliffe and P. D. Surry, "Fitness variance of formae and performance prediction," in *FOGA*, 1994, pp. 51–72.