Solving Graeco Latin Squares using MiniSAT

Vincent Tiono

15 October 2024

Table of Contents

- *Latin* Square
- *Graeco-Latin* Square

Introduction

- A *Latin* square of order n is an n x n matrix containing integer entries between 1 and n such that every row and every column contains each entry exactly once.

- For example, a 4 x 4 Latin square is shown below:

1	2	3	4
2	3	4	1
3	4	1	2
4	1	2	3

Problem Representation

- Encoding Latin square problem as a CNF.
- Variables and constraints.
  - Each cell can hold one of n values, so we have n^3 variables.
  - 3 constraints.

Constraints

1. Each cell contains exactly one number (1..n)
2. Each number appears exactly once in each row
3. Each number appears exactly once in each column

```
    for c in range(n):
    for v in range(n):
        # At least one number in each column
        clauses.append([varnum(r, c, v) for r in
            range(n)])
        # At most one number in each column
        for r1 in range(n):
            for r2 in range(r1 + 1, n):
                clauses.append([-varnum(r1, c, v),
                    -varnum(r2, c, v)])
```

Implementation: Three Steps

1. **Step 1: Problem Encoding**
   - Encode the *Latin* square according to its constraints.
   - Generate a corresponding CNF file called *latin_square.txt*.

2. **Step 2: Apply MiniSAT**
   - Using the command *minisat latin_square.txt latin_output.txt*, check satisfiablity.

3. **Step 3: Visualization**
   - Extract the solution and visualize the *Latin* square.

Results

1. Input Size vs Running Time and Memory Usage

Input Size (n)	**Run Time (s)**	**Memory Usage (MB)**
5	0.003	4.56
25	0.2	57.00
50	5.0	688.73
75	-	-
100	-	-

2. Maybe SAT isn't always the best approach. A simple alternative is shifting each row's numbers to the right!

1	2	3	4
4	1	2	3
3	4	1	2
2	3	4	1

MiniSAT with 46845000 clauses...

```
vincent_tiono@Vincents-MacBook-Air EDA % /Users/vincent_tiono/.pyenv/versions/3.12.6/bin/python /Users/vincent_tiono/Documents/EDA/latin.py
CNF constraints for a 75x75 Latin square written to latin_square.txt
vincent_tiono@Vincents-MacBook-Air EDA % minisat latin_square.txt latin_output.txt
============================[ Problem Statistics ]=============================
|                                                                             |
|  Number of variables:        421875                                         |
|  Number of clauses:        46845000                                         |
|  Parse time:                   4.05 s                                       |
|  Simplification time:         29.32 s                                       |
|                                                                             |
============================[ Search Statistics ]==============================
| Conflicts |          ORIGINAL         |          LEARNT          | Progress |
|           |    Vars  Clauses Literals |    Limit  Clauses Lit/Cl |          |
===============================================================================
|       100 |  421875 46845000 94921875 | 17176500      100    282 |  0.000 % |
|       250 |  421875 46845000 94921875 | 18894150      250    304 |  0.000 % |
|       475 |  421875 46845000 94921875 | 20783565      475    334 |  0.000 % |
|       812 |  421875 46845000 94921875 | 22861921      812    354 |  0.000 % |
|      1318 |  421875 46845000 94921875 | 25148113     1318    410 |  0.000 % |
|      2077 |  421875 46845000 94921875 | 27662925     2077    417 |  0.000 % |
|      3216 |  421875 46845000 94921875 | 30429217     3216    445 |  0.000 % |
|      4924 |  421875 46845000 94921875 | 33472139     4924    638 |  0.000 % |
|      7486 |  421875 46845000 94921875 | 36819353     7486   1479 |  0.000 % |
|     11330 |  421875 46845000 94921875 | 40501288    11330   2059 |  0.000 % |
|     17096 |  421875 46845000 94921875 | 44551417    17096   2409 |  0.000 % |
|     25745 |  421875 46845000 94921875 | 49006559    25745   2458 |  0.000 % |
|     38719 |  421875 46845000 94921875 | 53907215    38719   3190 |  0.000 % |
|     58180 |  421875 46845000 94921875 | 59297936    58180   3168 |  0.000 % |
|     87372 |  421875 46845000 94921875 | 65227730    87372   3310 |  0.000 % |
|    131161 |  421875 46845000 94921875 | 71750503   131161   3324 |  0.000 % |
|    196845 |  421875 46845000 94921875 | 78925553   196845   3518 |  0.000 % |
|    295371 |  421875 46845000 94921875 | 86818108   295371   3673 |  0.000 % |
|    443160 |  421875 46845000 94921875 | 95499919   443160   3701 |  0.000 % |
|    664843 |  421875 46845000 94921875 | 105049911   664843   3826 |  0.000 % |
|    997368 |  421875 46845000 94921875 | 115554902   997368   3942 |  0.000 % |
===============================================================================
INDETERMINATE
vincent_tiono@Vincents-MacBook-Air EDA %
⚠ 0   0
```

Indeterminate!

Graeco-Latin Square Introduction

- A *Graeco-Latin* square is a superposition of two *Latin* squares.
- Example of a 4 x 4 Graeco-Latin square:

$(1,2)$	$(4,4)$	$(2,3)$	$(3,1)$
$(4,1)$	$(1,3)$	$(3,4)$	$(2,2)$
$(3,3)$	$(2,1)$	$(4,2)$	$(1,4)$
$(2,4)$	$(3,2)$	$(1,1)$	$(4,3)$

Problem Representation: Graeco-Latin Square

- Encoding *Latin* square problem as a CNF.
- Variables and constraints.
  - n^3 variables for two *Latin* squares.
  - 3 constraints.

Constraints for Graeco-Latin Squares

1. Each cell contains a unique pair of values (1..n).
2. Each pair appears exactly once in each row.
3. Each pair appears exactly once in each column.

Graeco-Latin Implementation: Three Steps

1. **Step 1: Problem Encoding**
   - Encode the *Graeco-Latin* square according to its constraints.
   - Generate a corresponding CNF file called *graeco_latin_square.txt*.

2. **Step 2: Apply MiniSAT**
   - Using the command *minisat graeco_latin_square.txt graeco_latin_output.txt*, check satisfiablity.

3. **Step 3: Visualization**
   - Extract the solution and visualize the *Graeco Latin* square.

Results: Graeco-Latin Square

1. Input Size vs Running Time and Memory Usage

Input Size (n)	**Run Time (s)**	**Memory Usage (MB)**
3	0.003	4.56
5	0.02	6.71
7	97.57	135.44
8	8220	1069.96
10	-	-

2. Again, it can be done by shifting pairs to the left and right...

$(1,1)$	$(2,2)$	$(3,3)$	$(4,4)$
$(4,2)$	$(1,3)$	$(2,4)$	$(3,1)$
$(3,3)$	$(4,4)$	$(1,1)$	$(2,2)$
$(2,2)$	$(3,1)$	$(4,2)$	$(1,3)$

First, Let's Celebrate n=8!

```
~/documents/EDA
minisat graeco_latin_square.txt graeco_latin_output.txt
============================[ Problem Statistics ]=============================
|                                                                             |
|  Number of variables:          1024                                         |
|  Number of clauses:          157824                                         |
|  Parse time:                   0.03 s                                       |
|  Eliminated clauses:           0.00 Mb                                      |
|  Simplification time:          0.25 s                                       |
|                                                                             |
============================[ Search Statistics ]==============================
| Conflicts |          ORIGINAL         |          LEARNT          | Progress |
|           |    Vars  Clauses Literals |    Limit  Clauses Lit/Cl |          |
==============================================================================
|       100 |     949   110184   604986 |    40400      100     43 |  0.000 % |
|       250 |     949   110184   604986 |    44440      250     34 |  0.001 % |
|       475 |     949   110184   604986 |    48884      475     31 |  0.002 % |
|       812 |     949   110184   604986 |    53773      812     28 |  0.000 % |
|      1318 |     949   110184   604986 |    59150     1318     29 |  0.002 % |
|      2077 |     949   110184   604986 |    65065     2077     30 |  0.001 % |
|      3216 |     949   110184   604986 |    71572     3216     27 |  0.000 % |
|      4924 |     949   110184   604986 |    78729     4924     28 |  0.001 % |
|      7486 |     949   110184   604986 |    86602     7486     28 |  0.001 % |
|     11330 |     949   110184   604986 |    95262    11330     29 |  0.000 % |
|     17096 |     949   110184   604986 |   104789    17096     29 |  0.000 % |
|     25745 |     949   110184   604986 |   115268    25745     28 |  0.000 % |
|     38719 |     949   110184   604986 |   126795    38719     29 |  0.001 % |
|     58180 |     949   110184   604986 |   139474    58180     30 |  0.000 % |
|     87372 |     949   110184   604986 |   153421    87372     30 |  0.000 % |
|    131161 |     949   110184   604986 |   168764   131161     30 |  0.000 % |
|    196845 |     949   110184   604986 |   185640    44889     27 |  0.001 % |
|    295371 |     949   110184   604986 |   204204   143415     26 |  0.000 % |
|    443160 |     949   110184   604986 |   224625   103759     32 |  0.000 % |
|    664843 |     949   110184   604986 |   247087   118813     27 |  0.001 % |
|    997368 |     949   110184   604986 |   271796   219437     26 |  0.000 % |
|   1496156 |     949   110184   604986 |   298976   207179     30 |  0.000 % |
|   2244338 |     949   110184   604986 |   328873   102303     30 |  0.000 % |
|   3366612 |     949   110184   604986 |   361760   285763     28 |  0.000 % |
|   5050023 |     949   110184   604986 |   397937   232637     27 |  0.001 % |
|   7575139 |     949   110184   604986 |   437730    96305     27 |  0.001 % |
|  11362814 |     949   110184   604986 |   481503    86250     26 |  0.001 % |
|  17044326 |     949   110184   604986 |   529654   181712     27 |  0.001 % |
|  25566595 |     949   110184   604986 |   582619   482774     30 |  0.002 % |
|  38349998 |     949   110184   604986 |   640881   233771     31 |  0.000 % |
|  57525103 |     949   110184   604986 |   704969    55326     22 |  0.001 % |
|  86287761 |     949   110184   604986 |   775466   570114     25 |  0.001 % |
| 129431749 |     949   110184   604986 |   853013   490237     26 |  0.001 % |
| 194147731 |     949   110184   604986 |   938314   786825     26 |  0.001 % |
| 291221704 |     949   110184   604986 |  1032146   182181     26 |  0.001 % |
==============================================================================
restarts              : 393214
conflicts             : 344228102      (41873 /sec)
decisions             : 846810550      (0.00 % random) (103010 /sec)
propagations          : 8168286088     (993628 /sec)
conflict literals     : 9264502786     (14.81 % deleted)
Memory used           : 1069.96 MB
CPU time              : 8220.67 s

SATISFIABLE
```

```
Graeco-Latin Square (Combined):
[['(1, 5)' '(6, 3)' '(4, 2)' '(2, 6)' '(8, 7)' '(7, 8)' '(5, 1)' '(3, 4)']
 ['(6, 4)' '(1, 7)' '(2, 8)' '(8, 5)' '(5, 6)' '(3, 1)' '(7, 2)' '(4, 3)']
 ['(7, 3)' '(3, 5)' '(1, 1)' '(5, 7)' '(4, 8)' '(8, 4)' '(6, 6)' '(2, 2)']
 ['(5, 2)' '(2, 1)' '(7, 5)' '(6, 8)' '(3, 3)' '(4, 7)' '(1, 4)' '(8, 6)']
 ['(8, 1)' '(7, 6)' '(6, 7)' '(3, 2)' '(2, 4)' '(5, 3)' '(4, 5)' '(1, 8)']
 ['(4, 6)' '(5, 8)' '(8, 3)' '(7, 4)' '(1, 2)' '(2, 5)' '(3, 7)' '(6, 1)']
 ['(2, 7)' '(4, 4)' '(3, 6)' '(1, 3)' '(7, 1)' '(6, 2)' '(8, 8)' '(5, 5)']
 ['(3, 8)' '(8, 2)' '(5, 4)' '(4, 1)' '(6, 5)' '(1, 6)' '(2, 3)' '(7, 7)']]
```

Adding Pre-assigned Values, More Fun!

```
pre_assigned = [(1, 1, 1, 2), (2, 2, 1, 3)]
```

```
Graeco-Latin Square (Combined):
[['(1, 2)' '(4, 4)' '(2, 3)' '(3, 1)']
 ['(4, 1)' '(1, 3)' '(3, 4)' '(2, 2)']
 ['(3, 3)' '(2, 1)' '(4, 2)' '(1, 4)']
 ['(2, 4)' '(3, 2)' '(1, 1)' '(4, 3)']]
```

Future Research

- Higher-order combinatorial designs: magic squares, Sudoku-like puzzles, block designs, or Hadamard matrices
- Generalized combinatorial designs: transversal designs, tournament designs, or Steiner systems.
- Cryptographic applications: secret sharing schemes or encryption algorithms
- Latin Square Extension: Keen, Towers, Unequal. Credit to 黃思維!