SAT application - Graeco-Latin Square

This program is to find a orthogonal mate for a given latin square.
Some sample benchmarks are provided.

Graeco-Latin Square

A Graeco-Latin square is a pair of latin square which are orthogonal. i.e., the ordered paired entries in the positions are all distinct.

Usage

- Enter **python3 ./finder.py [input file] [output file]**

Concept

- For every row and column each of the n-digit appears exactly once
- Each pair of $(X_{r,c,d_1}, Y_{r,c,d_2})$ appears exactly once, where $1 \le r,c,d_1,d_2 \le n$,
  X is the given Latin square, and Y is the target Latin square.

Implementation

- Implement the first concept is quite easy, no further explanation is needed
- For the second concept
  - At least one : $(X_{r_1,c_1,d_1} \land Y_{r_1,c_1,d_2}) \lor (X_{r_1,c_2,d_1} \land Y_{r_1,c_2,d_2}) \lor \ldots \lor (X_{r_n,c_n,d_1} \land Y_{r_n,c_n,d_2}), \forall d_1, d_2$
  - At most one : After apply demorgan’s law,
    $((\overline{X_{r_1,c_1,d_1}} \lor \overline{Y_{r_1,c_1,d_2}}) \lor (\overline{X_{r_1,c_2,d_1}} \lor \overline{Y_{r_1,c_2,d_2}})) \land ((\overline{X_{r_1,c_1,d_1}} \lor \overline{Y_{r_1,c_1,d_2}}) \lor (\overline{X_{r_1,c_3,d_1}} \lor \overline{Y_{r_1,c_3,d_2}}))\ldots$
  - Since it’s not a **CNF**, apply **Tseitin Transformation** to make SAT solver be able to solve it.
- Using **Picosat** from **Pyeda** to solve the cnf.

Problem

Since **Tseitin Transformation** is needed, and the number of clauses is gigantic, so it will cause stackoverflow when the order of given Latin square is greater than 4.
So, it’s only able to find a orthogonal mate of a Latin square with order at most 4.

I’ve no idea how to fix it, and there’s very few information on finding orthogonal mate using SAT solver. Sadge