PURPOSE

This paper aims to develop and implement a set of 5 advanced techniques that encompass and summarize all known techniques, from the simplest to the most complex, such as:

- Naked Pairs, trios, etc.
- Hidden Pairs, trios, etc.
- X -Wing
- XY-Wing
- Swordfish
- Unique Rectangle
- Sole Candidate
- Unique Candidate
- Forcing Chain
- etc ...

Some of these 5 techniques are not intended to be a model of application by a person since, due to their complexity, they may only be available to computer processing.

These 5 techniques exclude brute force, trial and error, and backtracking techniques and, in general, any technique that is based on divination. Only techniques that allow solving a sudoku logic are used.

The design is designed to prioritize effectiveness over efficiency.

The implementation is designed with readability over efficiency.

The model can be improved to obtain more efficient results in terms of execution times, but it is not the objective of this paper to be efficient in execution time.

Sets have been used with their solutions, through the tool qqwing at the expert and intermediate levels. For intermediates, the efficiency is 100%. In the case of experts, the effectiveness is approximately 75% due to the fact that, in their set of tests, they are considered cases in which the application of brute force or divination techniques in general is necessary.

This paper addresses both the different techniques and the strategy to be applied.

The implementation has been carried out in Python using the Numpty, PyGame and Intertools libraries.

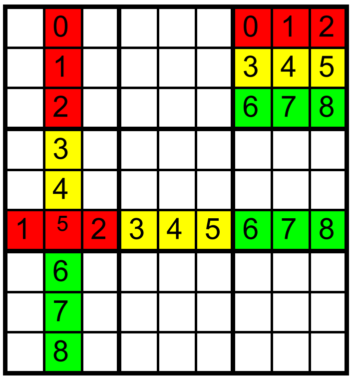
This paper has been translated by Google Translate. Sorry for mistakes!

TERMINOLOGY

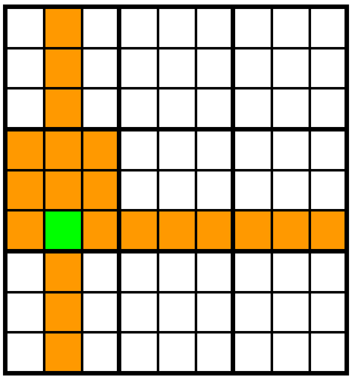
The structure of the sudoku is organized in 9 rows, 9 columns and 9 squares. Each group of 9 cells that make up a row, column or area will be called a "**tuple**" from now on, therefore considering a tuple as a collection of 9 enumerated elements. A sudoku is therefore a collection of 9 tuples.

1 row, 1 column, and 1 area are examples of the 3 types of tuples. In each tuple, are defined 3 numbered “**regions**”, grouping the 9 cells into groups of 3 consecutive cells according to their position.

In this scheme 3 tuples are represented: row 6, column 1 and area 2, with their cells numbered and highlighting the different regions of each (region 1 = red, region 2 = yellow, region 3 = green)



Are called **"neighbors"** to all those cells which shares with its own row, column and area. For example, the neighbors of the cell marked in green are the cells marked in orange. A cell always has 20 neighbors.



A tuple is formed by a value and a relation of 9 “**candidates”** that reflected as a matrix of 9 values formed by 0 and 1 where a 1 indicates that its position represents a candidate, being able to be in two possible states: either it has value or it has candidates

- value = 0 candidates = [0,1,1,0,0,1,0,0,1] -> empty cell with candidates 2,3,6 and 9
- value = n candidates = [0,0,0,0,0 , 0,0,0,0] -> cell with a defined value, without candidates

POSITION MATRIX REDUCTION

A position matrix is a matrix where n vertical elements are associated with horizontal m by means of marks. For example we can represent a problem in which we are told that there are 5 different people who have been given a numbered hat (from 1 to 5), but we do not know who has the hat of each number, they only give us a series of clues that allow us to fill a table with the possible values according to the clues. For example, if the clues were as follows, the matrix would be as:

- person 5 can only have a hat for
- hat No. 3 they can all have it except No. 5,
- person No. 2 has an odd hat,
- etc.

Let's say that after the clues we are left with this position matrix

	1	2	3	4	5
1		X	X	X	
2	X		X		X
3			X		
4	X		X		
5		X		X	

Since each person can only have a single shadow, and vice versa, the position matrix can reduce the number of cells marked based on the following rule:

> If a group of n cells on one axis have only the same number "n" of cells on the opposite axis (destination cells) associated with them, these associated values cannot be in none of the other source cells

So, we could deduce that as hat # 5 can only be held by the 2nd person, this person cannot have another hat, so we can remove the rest of cells

	1	2	3	4	5
1		X	X	X	
2					X
3			X		
4	X		X		
5		X		X	

Likewise, the same happens to hat # 1 now, only one person can have it, so this should be detached from any other hat.

	1	2	3	4	5
1		X	X	X	
2					X
3			X		
4	X				
5		X		X	

Lastly, hat # 3 can only be worn by one person, so we will also remove any excess marks.

	1	2	3	4	5
1		X		X	
2					X
3			X		
4	X				
5		X		X	

The rest of the positions are indefinite, without more clues we cannot continue deducing more.

These example reductions are level 1. Then there are level 2. For example, in the following matrix, hats 2 and 4 can only be in people 3 and 5, therefore these 2 people cannot have more hats than these 2, that is, 2 hats for 2 possible people.

	1	2	3	4	5
1	X		X		X
2	X				X
3	X	X		X	X
4			X		X
5	X	X	X	X	

By analogy, there are reductions of level 3, 4,5, ... always following the rule:

> If a group of “n” cells of an axis have only all of them associated as a whole with the same number “n” of cells on the opposite axis (destination cells), these associated values cannot be in any of the other source cells

An example Level 3 Reduction: Hats 2,4 and 5, including 3 can only be distributed to 3 specific men, therefore, these men cannot have any hat other than 2,4 or 5.

	1	2	3	4	5
1	X		X		
2	X			X	X
3	X	X		X	X
4	X		X		
5	X	X	X	X	

This system is specially applied to reduce the possible positions of the candidates in a tuple, generating a 9x9 position matrix where on an axis one represent the 9 cells of the tuple and on the other axis the 9 possible candidates, checking in each cell if it can contain each of the candidates.

For example, the tuple [6,3,0,0,1,2,0,0,0] could be represented as (cells on the left axis and candidates on the top axis): Note that the cells with values do not have candidates.

	1	2	3	4	5	6	7	8	9
1									
2									
3				X	X		X	X	X
4				X	X		X	X	X
5									
6									
7				X	X		X	X	X
8				X	X		X	X	X
9				X	X		X	X	X

In a position matrix, the Cells that already have values are not represented with marks, only the candidates, since the position matrix is intended to reduce the candidates. If there are cells with values, they are empty in the matrix.

When the reduction is applied for the 9 rows, 9 columns and 9 areas, and an improved matrix is obtained (with fewer marks), it means that in the tuple we can discard values of in certain cells. Reducing the position matrix of a tuple allows to advance in the resolution of this, and therefore, the sudoku in general.

Matrix reduction is also applied at a global level in which for each value from 1 to 9 a 9x9 position matrix is created (representing the structure of the sudoku) where it is represented if said value can exist in the cell. This reduction is called a global value reduction. Thus, each value from 1 to 9 can be represented in a position matrix reflecting its possible cells in the 9x9 box that represents the sudoku.

STRATEGY

All the techniques, except one, aim to reduce the number of possible candidates in each cell. The remaining technique aims to check if there are cells with a single possible candidate, and in such a case, to assume such as the value of the cell, and exclude it as candidate in all its neighboring cells.

To solve a sudoku, all the techniques are executed in random order, and at the end, if none of them has made any change in the sudoku, either by assigning values or discarding candidates, or if there are no empty cells, the system will conclude its resolution. Meanwhile, the system will re-apply this strategy cyclically, until either there are no more empty cells, or until after applying all the techniques, none have managed to make any changes.

It has not been possible to verify that there is a specific order of application of the strategies that offers better results in a sustainable way and independent of the difficulty of sudoku. In general, using a random sequence gives a better result.

In general, the first 3 techniques are much more powerful than the last 2, but by trying to solve following a sequence based on first using the most powerful techniques first, there have been no improvements in efficiency, although they have in efficiency.

In general, by symmetry, all the techniques are executed both on the original version and on the transposition of the current grid.

FIVE TECHNIQUES

FIND VALUES UNIQUE

When a matrix associated to a tuple, therea row or a column with a unique mark it in a position of the tuple can contain only one value, and / or a candidate can only be a single tuple position. This technique consists of reviewing the position matrices with the candidates of each tuple. Every time this situation is found, a value of one cell is assigned. This is the only technique that will assign values to cells based on only one candidate remaining.

For example, if we represent the position matrix of a tuple and obtain this example, we can deduce that we already have two cells with values (1 and 2 since they do not show candidates) and that we can deduce that we have 2 more cells that we can define their value (marked in green), since these only have a value in their row or column.

	1	2	3	4	5	6	7	8	9
1									
2									
3		X		X	X	X	X	X	X
4				X	X		X	X	X
5	X	X			X	X	X		
6		X							
7				X	X		X	X	X
8				X	X		X	X	X
9				X	X		X	X	X

Once the allocation of the 2 new cells has been applied, the position matrix of the tuple will be as follows, noting that after this process another value can be deduced again (column 6). This process then collects the cells that only have a value in their row or column and converts them into values of that cell, canceling their candidates so as not to represent them in their position matrix.

	1	2	3	4	5	6	7	8	9
1									
2									
3				X	X	X	X	X	X
4				X	X		X	X	X
5									
6									
7				X	X		X	X	X
8				X	X		X	X	X
9				X	X		X	X	X

MATRIX REDUCTION

Each tuple It can be represented as a position matrix where the positions of the cells are shown on one axis and the possible values it can take on another axis.

Using the matrix reduction technique, as was done in the example of people and hats, these reductions can be translated into candidate discards.

There are advanced techniques, such as XWING or SWORDFISH, hidden pairs, hidden threesomes, etc. that are partial examples of the implementation of this technique. In the end, all of them seek to reduce a position matrix to reach conclusions.

This technique, along with the following, are the most effective and powerful.

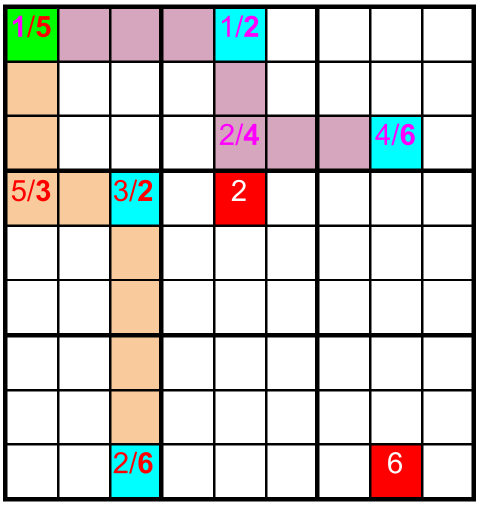
OVERLAPPING SCOPE

This technique represents the situation of “whatever you do in the end, such a cell will have such a value”, that is, finding cells with x candidates such that the one chosen is chosen, in a certain specific cell of the sudoku it will be concluded that It will have the same specific value independent of the value chosen in the starting cell. There is the technique called forced chains and XYWING which are a partial implementation of this technique.

When an unassigned cell in the sudoku has 2 or more candidates, you can try to form cell chains starting from this initial, and searching among its “neighbors” (cells of the same row, column or area) for others such that it has 2 possible candidates where one of the possible candidates coincides with one of the previous cell, thus forming a chain.

The strategy is to locate cells that, regardless of assuming any of the chains of the same cell, always have to discard the same candidates.

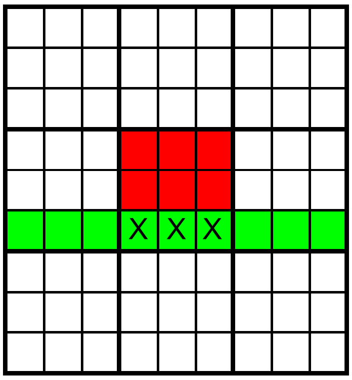
For example: if the starting cell (green) has two possible candidates, 1 and 5, and other cells have the candidates represented in the example, 2 strings could be generated (one starting with the value 1 represented by the purple cells and another starting with the value 5 represented by the orange cells). Therefore, choose the value you choose in the first cell (green), the red cells cannot have the value indicated in them. This means that the scope of the value 1 and the scope of the value 5 overlap in the red cells with the value 2 and 6 and therefore can be discarded as candidates in these red cells.



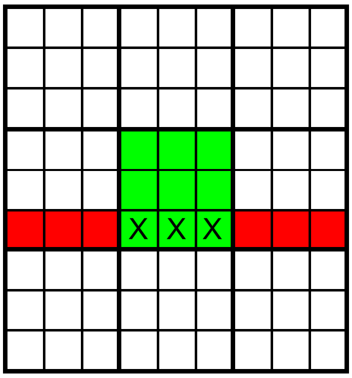
REDUCTION OF ASSOCIATED AREAS

When a row or column has to a certain value a set of defined possible cells in a single "region" of the tuple, that is, they share the same area, this value can be discarded from the other cells of that area that are not part of the tuple.

For example, if in a row (marked in green) the value V is missing and can only be in some of the 3 cells marked with "X", this value V can be discarded as a candidate for each cell of the area that is not part of the row (cells marked in red).

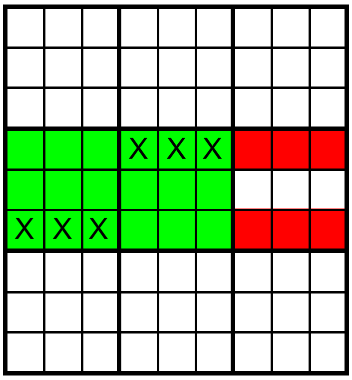


Analogously, the same is true in reverse: If a certain value can only be in one of the 3 regions of an area (marked with X), the cells of the row that intersects the area in that region cannot contain that value in the rest of the cells (marked in red).

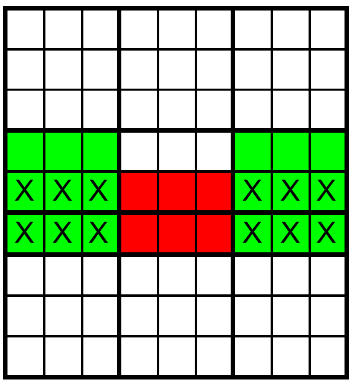


REDUCTION OF NEIGHBORING AREAS

If in a set of 3 adjoining areas, a candidate can only be in a different region in two of these areas, it will be possible to discard those two regions in the third area.



Likewise, if two of these areas can only contemplate the same two regions for a certain candidate, this candidate can be discarded from said two regions in the third area.



FILES

- **sudoku_class**.py
- **tupla_class**.py
- **cell_class**.py
- **techniques**.py
- **graphics**.py

REFERENCES

- https://qqwing.com/ for obtaining test sets
- http://trevorappleton.blogspot.com/2013/10/guide-to-creating-sudoku- solver-using.html as inspiration for the graphing module.