



## The Model and Algorithm to Estimate the Difficulty Levels of Sudoku Puzzles

Chungen Xu & Weng Xu

Department of Applied Mathematics, Nanjing University of Science & Technology

Nanjing 210094, Jiangsu, China

Tel: 86-25-8431-5877 E-mail: [xuchung@mail.njust.edu.cn](mailto:xuchung@mail.njust.edu.cn)

*The research is financed by the National Natural Science Foundation of China under Grant No. 70871059.*

### Abstract

Sudoku is a number placement mathematical puzzle based on logic. The purpose of this paper is to discuss suitable models and algorithm to generate Sudoku puzzles of varying difficulty. It is generally recognized that hand-made puzzles are more enjoyable than those generated by computer. Our goal is to establish models to generate Sudoku puzzles of varying difficulty, which are as enjoyable as hand-made ones. As we believe that puzzles generated by simulating the design process of hand-made ones will also be of much enjoyment, we established our first model -No Brute-Force. Brute-Force technique is excluded from this algorithm, for there is no enjoyment in solving puzzles using it. We have implemented the algorithm with a JAVA program. We conclude that it is possible and reasonable to generate Sudoku puzzles of varying difficulty as enjoyable as hand-made ones.

**Keywords:** Sudoku, Puzzle, Difficulty ratings, Complexity

### 1. Introduction

In history, Suduko, which is an interesting number placement puzzle that requires logic skills and patience, first appeared in the U.S. in 1979 (WIKIPEDIA, July 2008). The game was designed by Howard Garns, an architect who upon retirement turned to puzzle creation. In the 1980s, the game grew in popularity in Japan. It is a fantastic puzzle game that can be found in newspapers, books and even on the game websites nowadays. Its rules are extremely simple. The classic puzzle consists of a 9x9 grid that is divided into nine 3x3 blocks, with some cells already contain numbers, known as "givens". The goal is to fill the empty cells, so that each column, row, and blocks contains the number 1 through 9 exactly one.

It is true that nowadays, millions of websites offer Sudoku puzzles online, and several free generator software are available, but there are quantities of solvers prefer hand-made puzzles rather than computer-generated ones. The most important reason is that the computer-generated puzzles are not as enjoyable as crafted ones. For instance, it is a tiresome process to solve a puzzle either too difficult or exists more than one solution. However, as the group of Sudoku fans expanded quickly, it is impossible to design all the puzzles needed by hand. As a result, it is wrong to ignore the computer generators' advantages, such as efficiency. What's more, we should develop computer generators both in theories and techniques (ALBERTO, 2006. p16-21).

Theoretically, it is essential to develop an algorithm to generate Sudoku puzzles of varying difficulty, which are as enjoyable as those crafted ones. The algorithm should guarantee a unique solution. And it is necessary to minimize the complexity of the algorithm.

Difficulty rating is a complex topic, for it is a subjective process which is uneasy to make quantitative analyses.

In this paper, we have established a model to generate puzzles as enjoyable as possible, based on that they at least meet the classic Sudoku's basic requirements. The model has its own metrics to define a difficulty level. We have implemented the model with algorithms performed well in programming language JAVA, and we have analyzed the complexity of the algorithm in details.

Rating difficulty is a complex topic, for it is a subjective process. In order to develop an algorithm to generate puzzles of varying difficulty, we should develop metrics to define difficulty level. So, factors having nothing to do with difficulty ratings should not be considered. Difficulty levels ranked by computer should be consistent with those most of the Sudoku solvers' believed. Algorithms can take a factor related to difficulty rating into consideration only if that the factor can be

analyzed quantitatively.

The complexity of a solving technique required and how many times it has been used can be estimated by computer. This estimation allows computer generators either construct a Sudoku puzzle with certain difficulty level or to tailor their puzzles to audiences of varied solving experience.

Firstly, we construct a model simulating the design process of hand-made puzzles - No Brute-Force Model. It implements the other standard, and it never uses brute-force in generating a puzzle, thus, it guarantees the puzzles' enjoyment and reduces the complexity of algorithm.

## 2. No Brute-Force Model

### 2.1 Foundations

#### a) Number of givens

The Japanese publisher Nikoli limits its Sudoku puzzles to a maximum of 32 givens. One can conclude, however, that a Sudoku with more than 40 givens is probably not a very difficult one.

#### b) No Brute-Force

Brute-Force technique can deal with all the Sudoku puzzles, even those of more than one solution. However, it is a technique for computer rather than people. Moreover, there is no enjoyment in solving Sudoku puzzles using Brute-Force technique.

The probability that a Sudoku puzzle generated by computer randomly demands Brute-Force technique is not small. Thus, if we want to exclude those puzzles, we have to solve the puzzle using techniques without Brute-Force in advance. If the puzzle is happened to be a puzzle demands Brute-Force, we have to generate a new puzzle.

#### c) Solving techniques

In this model, our algorithm also only implements these techniques, such as:

Naked Single, Hidden Single, Naked Pair, Hidden Pair, Intersection Removal, Naked Triplet, Naked Quad, Hidden Triplet, Hidden Quad, X-wing, Swordfish, Jellyfish, XY-wing, XYZ-wing, WXYZ-wing and Forcing chain.

If a puzzle demands more techniques, we assume that it is too hard for people to solve. Our algorithm won't generate puzzles like that for our goal is to develop an algorithm to generate Sudoku puzzles of varying difficulty, which are as enjoyable as hand-made ones.

Codes of these solving techniques reference the source code of "Sudoku Explainer", a free generator whose source code is available, as well as the first mode(SANTA, Dec 28, 2007).

### 2.2 Development

#### a) Generate a grid with some givens.

Begin with an empty grid, and then generate a puzzle take advantage of the first algorithm, only use Naked Single and Hidden Single techniques. If the difficulty level needed is easy, the puzzle has been generated successfully.

#### b) Remove cells one by one as much as possible.

We have explained that the number of givens has little or no bearing on a puzzle's difficulty. But solving a puzzle with more than 40 givens is not an enjoyable process, for it provides little space for us to exert.

Our algorithm is designed to remove givens one by one as much as possible until the puzzle can't guarantee a unique solution. Then put the last cell back, and try another one until all the givens have been tested.

c) Make difficulty ratings on a puzzle and generate a puzzle with needed difficulty. In this algorithm, we make difficulty ratings with another standard different with the first model. There is a value represent the difficulty for each solving technique. Furthermore, considering that it is more difficult to use a technique at the first time than others (see Table 2), we set a higher value for the first use of a certain technique. We are inspired by a discussion in a forum. (HOWARD, July 31, 2005).

In order to make difficulty ratings, we have done a statistics about the puzzles' difficulty level provided by newspapers (see Table 1). Although there are a small number of samples, we consider that it can be applied in our models.

Here are the results: see Table 2. Therefore, we have made 4 levels as Table 3.

Rate the puzzle generated to test its difficulty level. If the puzzle's difficulty level is consistent with the requirement, the puzzle has been generated successfully. Otherwise, repeat from step 1.

#### d) Flow diagram of the algorithm.

Flow diagram of the algorithm see Figure 1.

## e) Implementation.

We implement the program in Eclipse 3.3.0 with the programming language JAVA. Codes of solving techniques reference the source code of "Sudoku Explainer", a free generator whose source code is available (SANTA, Dec 28, 2007).

### 3. Conclusions

The model we established has developed an algorithm to generate Sudoku puzzles of varying difficulty, which are as enjoyable as hand-made ones. No Brute-Force Model, without Brute-Force technique, is also an excellent generator with the ability to produce enjoyable puzzles of which difficulty level are consistent with the level needed. The metric to define a difficulty level is different from others and is reasonable. Further work we have considered about our model is to adjust them to be fit for other forms of Sudoku puzzles.

### References

- Alberto, Moraglio, Julian, Togelius and Simon, Lucas. (2006). *Product Geometric Crossover for the Sudoku Puzzle*. In 2006 IEEE Congress on Evolutionary Computation. July 2006. p. 16-21.
- Dai, Min. (2006). *Data Structure Courseware*. Tianjin University of Technology. 2006. p. 8.
- David, Bau. (2006). Sudoku Generator. [Online] Available: [http://davidbau.com/archives/2006/09/04/sudoku\\_generator.html](http://davidbau.com/archives/2006/09/04/sudoku_generator.html). September 04. 2006.
- Frazer, Jarvis. (2008). Sudoku enumeration problems. [Online] Available: <http://www.afjarvis.staff.shef.ac.uk/sudoku/>, February. 2008.
- Huang, Jing. (2006). *iSudokuHelp*. A-Crystal-Man Software Studio, 2006.
- Howard. (2005). Rating Difficulty. [Online] Available: <http://www.setbb.com/sudoku/viewtopic.php?t=142&mforum=sudoku>. Jul 31, 2005.
- JBL. (2008). *Difficulty ratings*. [Online] Available: <http://www.spiritustemporis.com/sudoku/difficulty.html>.
- Santa, Clara. (2007). Sudoku Explainer. [Online] Available: <http://diuf.unifr.ch/people/juillera/Sudoku/Sudoku.html>. Dec 28. 2007.
- Wikipedia: Sudoku. (2008). [Online] Available: <http://en.wikipedia.org/wiki/Sudoku> (July ,2008).
- Lee, Wei-Meng. (2006). *Programming Sudoku*, Apress. 2006. p.146.

Table 1. The first time than others

Techniques	First time	Others
Naked Single	1	1
Hidden Single	2	2
Naked Pair	5	3
Naked Triplet	6	4
Naked Quad	8	5
Intersection Removal	8	5
Hidden Pair	10	6
Hidden Triplet	13	7
Hidden Quad	16	8
X-wing	10	6
Swordfish	16	8
Jellyfish	18	9
XY-wing	13	7
XYZ-wing	16	8
WXYZ-wing	20	10
Forcing chain	24	12

Table 2. The puzzles' difficulty level

Level	I	II	III	IV	V	VI	VII	VIII	IX	X	Range
Easy	62	66	58	50	69	58	53	50	71	65	≤71
Intermediate	79	88	83	86	85	76	83	81	86	74	74-88
Hard	84	89	95	97	98	85	96	94	89	88	84-98
Challenging	109	112	103	102	108	98	127	96	124	120	≥96

Table 3. 4 levels

Level	Range Easy
Easy	≤72
Intermediate	72-86
Hard	86-98
Challenging	>98

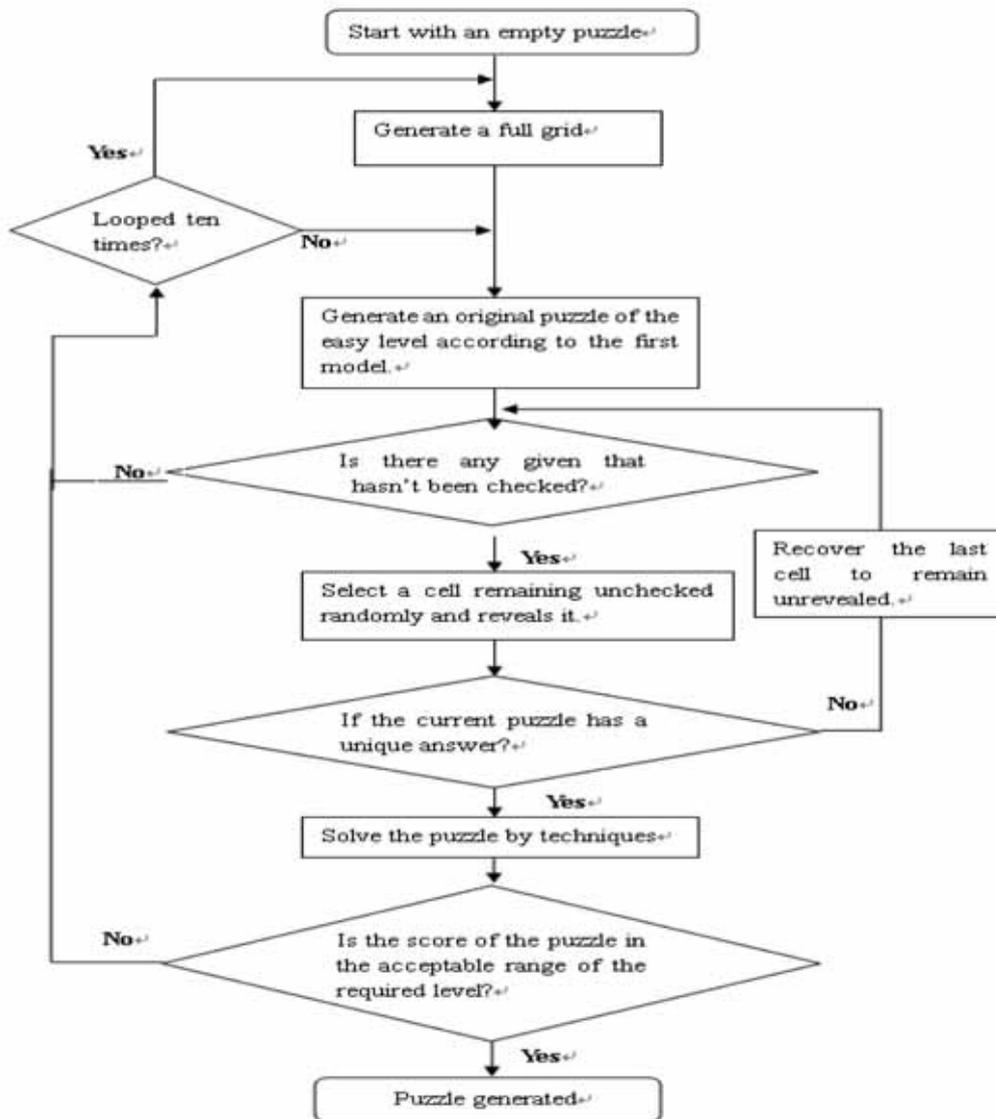


Figure 1. 1 Flow diagram of the algorithm