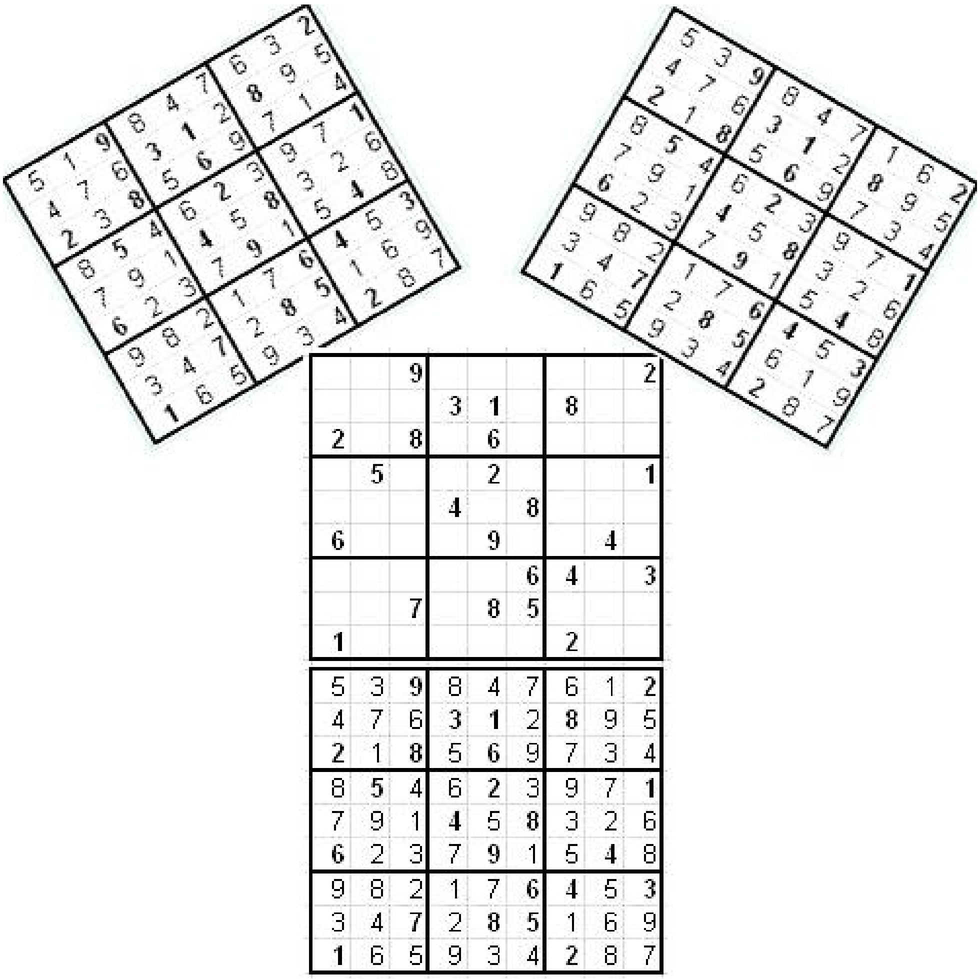
1Su Doku puzzles



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

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

Difficulty and solutions

Jacques PARIS

SuDoku puzzles : difficulty and solutions

Jacques Paris, march 2006

Asserting the difficulty level of a SuDoku puzzle requires knowing its solution and the procedures used for finding that solution. I am convinced from the small experience I have that puzzle "designers" have not all grasped the complexity of this problem and can be at times rather erratic in the way they attach to their puzzles labels like "easy", "hard", "devilish".

I would not have been able to undertake the explorations on which this paper is based without my "Template for help to solving Sudoku puzzles" (see appendix for details). This small Excel worksheet keeps track of the potential assignations i.e. the integers that can be assigned to any position other than clues and already made choices while respecting the basic SuDoku rules. It does not do any assignation by itself, it is a simple electronic accounting tool that gives the opportunity to try alternative solutions and see the way they develop.

A measure of difficulty.

I have named one element of the "Template…" the "Global Performance Table", GPT in short. That 9x9 table indicates for each position how many different integers could be assigned to it. A 0 indicates a clue or an already assigned cell. A 1 shows that there is only one candidate integer for that position; that clue must me respected and should disappear in an automatic assignation phase. Larger integers (2 to theoretically 9, but I have not found yet any value above 7) reveal then the numbers of candidates.

If the value is 2, we have first to find which integers are involved and let us name them M and N. There are two ways to look at them: as potential "branches" of a decision tree – a branch describes the consequences of having selected M or N and every branch is conserved –, or as attempts to reach as solution – if M leads to a solution it is accepted, if not it must be N –. This can be easily extended to values higher than 2.

I prefer the second view that reminds me of my training in statistical analysis; it is indeed very similar to the notion of "degrees of freedom" (dof) that indicates how many "free" choices can be made before a situation is fully described. I have thus proposed as a general measure of difficulty the sum of all the "non-null values minus one” in a GPT; an alternative formulation would be the sum of all the values in a GPT minus the number of non-null values.

A dof thus calculated is a global indication of the complexity of the situation (the number of all free choices taken one position at the time) given the available information (the already assigned cells and the consequences of these assignations following basic SuDoku rules implementation). The higher is the dof, the more numerous are the free choices, the less constrained is the situation.

Here is an example of a puzzle (left), its GPT (right) and its dof (center) detailed as a frequency distribution of the GPT. 27 (0) is the number of clues and 54 (bottom line) is the number of non-zero values or of empty positions. There are no 1s in the table and that shows that the GPT has detected no automatic assignation (some may be revealed by other means). The dof of 135 is the sum of (10*1 + 18*2 + 18*3 + 5*4 + 3*5)

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

0	27
1	0
2	10
3	18
4	18
5	5
6	3
7	0
54	135

3	5	0	0	4	4	4	6	3
0	4	2	2	3	0	0	4	0
0	4	3	3	0	4	2	4	0
3	4	0	0	3	2	0	4	0
3	5	3	4	0	4	4	6	3
0	4	0	3	2	0	0	3	2
0	4	3	3	0	3	4	6	0
0	5	0	0	2	2	2	3	0
2	4	3	5	4	0	0	5	3

Fig. 1

Determinism and choice.

When confronted with a new SuDoku puzzle, one should start by applying the basic rules and find out where that operation leads. This phase is described in “Template…” as “getting rid of the ones”, a 1 in specific areas of the templates specifying a required assignation. That phase is sometimes referred as a deterministic solution in the sense that there are no free choices exerted, only mechanical recordings of constraints.

Very often, this phase will end up with a solution, i.e. the entire puzzle can be filled up. This is the D family of SuDoku puzzles, D for direct or determined.

But sometimes the deterministic phase ends up (no more ones in the “Template…” critical areas) before the puzzle is solved. One must thus exercise a choice, decide which value to try in some empty position. That choice must be made of course among the potential integer candidates for that position.

A choice made, there should be a new automatic assignation phase ending possibly into a dead end requiring a new choice to be made, until a solution is found. I will name the combination of a choice and its automatic consequences a step (it takes us a step closer to the solution) and a puzzle that cannot be solved automatically a stepped puzzle; hence the S family of puzzles.

Seen under that light, a choice during a SuDoku puzzle solving process is made at specific points; it is clearly expressed as the selection of one integer in one position and any other intervention or choice is held back until all the consequences are registered. Any choice is constrained but it can also be guided; it is the purpose of the "Template…".

Difficulty level of D-puzzles

A directly solved puzzle cannot be really named difficult because it contains all the information required for solving it. It may however offer at any time more or less indications of possible assignations and that may give a feeling of lesser or greater difficulty. If one can get an early "impression" when looking at a new puzzle, this feeling is often reinforced or modified during the process; it is not rare to find sequences where the last identifiable assignation may reveal only a new one, and such a chain of one-by-one revealed assignations weighs probably much more in increasing difficulty than sequences where several possible assignations are revealed at the time.

Number of clues

Among the parameters that are available from the start is the number of clues (it is assumed that the more clues are given, the easier is the puzzle); that measure is the most objective of all and every one can agree on it. But is it really the solid basis for establishing difficulty level?

> We must give here a precision: a measure is some quantified parameter; a difficulty level is a scale of a limited number of levels each with its own "label", into which a measure is converted. Clues could range theoretically from 1 to 80. Difficulty scales have different number of levels and their labels reveal increasing difficulty such as "Easy", "Average", "Hard", "Devilish".

I have analyzed all the puzzles offered in a book [1] to find 81 D-puzzles.

[1] "100 Su Doku no. 1" by Wayne Gould, (Éditions Générales First, Paris, 2005) originally published as "The Times Su Doku Book 1" (Harper Collins Publishers)

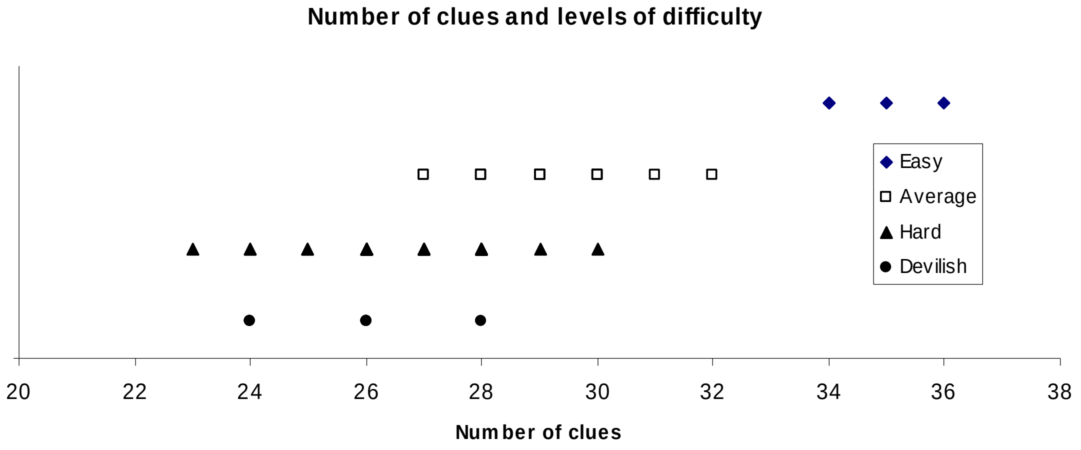


Fig. 2

Fig. 2 shows all the clue values scattered by level of difficulty (there are much fewer distinct points than the 81 original because several puzzles have the same number of clues). If there is an apparent relationship between difficulty level and number of clues, it seems to hold only for the first 2 levels; some overlap exists between Average and Hard and absolutely no distinction can be made between Hard and Devilish.

Automatic assignations

If one could imagine that the GPT could provide an interesting way to measure initial difficulty, he will not go far. This measure would be the number of ones in that table but this number ranges from zero to around 10 (very close probably to the real but unknown upper limit) and zero can be found in almost every level. If one would try to include all the ones in the specific template areas, he will run the risk of double counting some entries detected in more than one specific area.

GPT Degrees of freedom

That measure already explained may be the most sensitive one. Let us see what the performance of the 81 puzzles with that measure is (fig. 3).

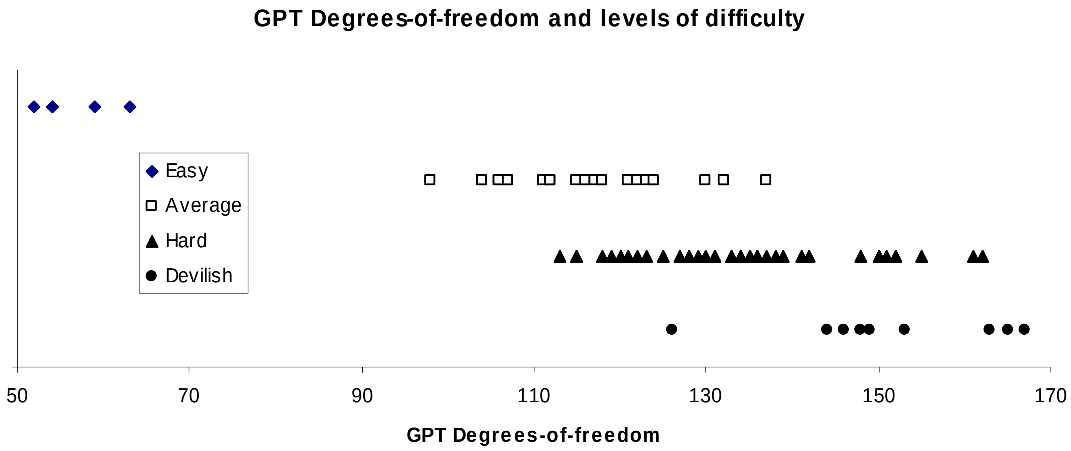


Fig. 3

This is a finer measure that the number of clues, it gives more distinct values. The general relationship between GPT-dof and the author's levels of difficulty is more established. If there are overlaps between the hardest 3 levels, they are not as global as with the number of clues. One can also imagine the possibility of using breakpoints detectable in the scatters to suggest a new classification scheme such as this 5 level scale: below 90, 90-110, 110-130, 130-150, above 150. But I do not want to enter here that much too subjective operation of scale building (defining breakpoints and labels) because we have covered yet all possible types of puzzles.

One-S and Multiple-S types of puzzles

A puzzle that cannot be solved deterministically in one phase was named an S-puzzle, a stepped puzzle. If all the S-puzzles share that characteristic, they differ widely by another feature, the number of steps required to solve them. That difference has so much impact as we are going to see that we propose to create two sub-families, the SS- and XS-puzzles.

Single step SS puzzles are simply detected by making a choice between the two integers that are candidates for a position. If that "step" (choice + automatic assignations) leads to a solution, we have a SS-puzzle. If it fells short of a solution, it is a MS-puzzle. There is also the possibility that it reaches an incompatibility, i.e. the consequences of the choice break the basic rules; if that case, trying the alternate candidate will lead to an acceptable outcome (remember that one of the two candidates must be part of the solution), a final solution (SS-type) or an incomplete one (MS-type).

I may not have stressed enough the fact that the assignations obtained by the applications of the basic rules (automatic or deterministic assignations) yields a unique solution, the procedure being completely independent of the order in which the assignations are made. D-puzzles have a unique solution; SS-puzzles also because they are the combination of two "unique outcome" operations, the initial one and the one following the choice defining the only step.

That notion that a SS-puzzle is in fact formed of two puzzles nested one into the other lead me to consider adding to the initial dof value that obtained at the end of the first automatic assignations phase, the initial dof of the first and only step in order to create a coherent and continuous measure. I will explore that possibility later, after dealing with MS-puzzles.

The existence of a single solution for a MS-puzzle cannot be ascertained because acceptable alternatives (partial solutions) can exist at any step and cannot a priori been thrown away. I will not go to saying that MS-puzzles cannot have a unique solution but after the presentation of an example of MS it will be obvious that the probability of such outcome is probably nil.

Approaches to solving MS-puzzles

I want first to justify my choice of working only with choices involving 2 integers in a position, as identified in the global performance table by the 2's. Dealing with simply two possible values for a given position may seem to be a restrictive choice; why not more than two in a position, or one integer in two positions (as indicated in the totals areas of the integer performance tables) and even more possibilities. I have restricted my search to the cases of 2 integers for 1 position for a question of efficiency certainly but also for a theoretical reason: the chosen integer must be part of the final solution and as the order of definition has no import on the final outcome, starting with the most exacting constraint (one out 2 rather than out of 3 or more) is expeditious and does not eliminate acceptable solutions.

One can visualize a MS-puzzle solving context as a descendant genealogy tree. The head of it is the puzzle as defined after the initial automatic assignations. A branching out corresponds to a step. The descendants are the outcomes of the various "2 for 1" choices identifiable in the GPT. These outcomes are of 3 kinds as we have already seen: final solution (end of line descendant), partial solution initiating a new step (new branching out) or impossibility (no descendant). But an image has its limitations and in the case of MS-puzzles one should take into account two possibilities not present in family trees.

A choice may have no impact by not revealing any new 1 in the “Template…” critical areas; no automatic assignation is possible in that case. I would tend to discard these outcomes has of no practical import.

The greater difference with a descendant tree is that some outcomes at any given level may be identical. I will treat these “homozygote” siblings as a single outcome, the purpose being to find possible differing outcomes (and solutions) and not the number of different ways a particular pattern can be generated.

With these preliminaries well established, let us look in the two approaches I can imagine to puzzle solving. One I will refer as “exhaustive” considers all possible combinations and build down all the tree branches until their end (final solution). Such approach will identify every possible solution to the puzzle and requires generous computing and memory capacity. The other is much more “selective” and is aimed at finding one solution with the minimum amount of investment. Both approaches will be exemplified by a demonstration on the same puzzle, the famous (in my one eyes at least) #88 in the Gould’s book (see footnote1).

“Exhaustive” solving of a MS-puzzle

The proposed puzzle is presented in Fig. 4

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

0	**25**
1	**0**
2	**5**
3	**16**
4	**17**
5	**16**
6	**2**
7	**0**
56	162

5	5	0	0	5	5	3	3	4
3	4	2	4	0	0	0	3	0
6	5	5	4	5	5	0	4	5
3	0	3	4	4	5	2	0	0
3	0	0	4	2	3	0	0	3
0	0	3	4	4	5	3	0	2
5	3	0	4	0	4	4	3	5
0	4	0	0	0	3	3	2	5
5	3	4	4	4	0	0	5	6

Fig. 4 The #88 original puzzle

After automatic assignations, it looks like in Fig. 5

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

0	38
1	0
2	8
3	15
4	17
5	3
6	0
7	0
43	101

4	4	0	0	4	5	3	3	4
2	3	0	4	0	0	0	2	0
4	3	0	4	4	5	0	4	4
0	0	0	0	2	2	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	2	3	3	2	0	2
3	3	0	4	0	4	3	3	4
0	4	0	0	0	3	3	2	5
3	3	0	4	3	0	0	4	4

Fig. 5 #88 after automatic assignations, ready for step 1

The way to proceed is to identify the integer pairs in each position where a 2 appears in the GPT (left part of fig. 5). The situation can be described in a table such as Fig. 6. The “choices” are named by row, column and value (21.2 means in row 2 column 1, value 2); the DOF are those calculated at the end of the automatic assignations (a “=” indicates that the choice had no impact on further automatic assignations); the outcomes identify the different patterns that can be identified following the completion of that step

Choice	DOF	Outcome
21.2	=	
21.5	92	a
28.3	92	a
28.5	78	b
45.5	94	c
45.9	94	d
46.5	94	d
46.9	94	c
64.3	90	e
64.6	=	
67.4	95	f
67.8	65	g
69.4	65	g
69.8	95	f
88.4	85	h
88.5	78	i

Fig. 6 Outcomes from step 1

There were 8 two’s in the GPT, that is 16 different choices. Only 9 different patterns (a through i) have emerged (2 without impact, 5 duplicates). The purge of identical patterns is an extra operation that requires special attention but that pays off nicely by pruning the tree of unnecessary branches.

I have worked out to the end only one branch formed by the first (the "a") pattern identified at each step. Rather than showing all the details for each step, I have summarized some characteristic in the table of Fig. 7.

step	choices	patterns	Dof range	=	Impos.	duplicates	solutions
1	16	9	65-95	2		5	
2	18	11	62-86	1		6	
3	22	12	8-57	2		8	
4	20	14	8-57	2		4	
5	30	17	14-36			13	
6	30	7+2	4-30			13+8	A, B
7	28	2+3	8-16			11+12	A, C, D
8	32	0+2	-			32	A, D

Fig. 7 performance along the branch "first pattern"

The number of choices for one step is equal to the sum of

- the different patterns differentiated between partial and final solution (step 6, 7 patterns to be "continued" + 2 final solutions)
- the choices without impact (=)
- the choices leading to an impossibility
- the duplicates differentiated between patterns to be continued and final solutions (step 6, 13 duplicated over the 7 patterns to be continued +0 8 over the 2 final solutions)

"Dof range" is the range of variation of the dof values for the patterns to be continued. One can see that the maximum has a tendency to get steadily smaller but not the minimum; it is not too surprising because the range of step N+1 is not based on the smallest value of step N but on one among those in the range of that step. This is one specific feature of the "exhaustive" approach.

The last column identifies the different final solutions unveiled at each step. Several observations can be made with only these partial results. The first will be that there is no way to predict the length of the branches that is the number of steps to exhaust all possibilities. No final solutions are encountered before step 6 along that branch and two more steps are required to complete the branch. I believe that forgetting the "no impact" choices makes the tree more compact without losing possibly different solutions.

The same solution can appear in different branches; the same pattern can be defined as a final outcome of noticeably different "paths" or decision sequences. It is possible that a pattern recognized in a step be the same as one identified in a previous step; as it is not eliminated by the present procedure, it is not surprising that the same solution could be reached via both routes. A more sophisticated pattern retention algorithm would compact the tree perhaps noticeably.

Before moving to the other approach, I solved another “twig”; instead of choosing pattern “a” from step 7, I chose its neighbour, pattern “b”. That new step 8 offers 16 choices leading to 2 final solutions C and D. If one was to make a typological mapping of the final solutions on the complete tree, it will most probably observe that they have a tendency to spatially regroup revealing the basic choices that made them different.

“Selective” solving of a MS-puzzle

The selective approach is based on the notion that a puzzle with a lower dof value is closer to a final solution than one with a higher dof. The implementation of that assumption is straight forward. At each step, the choice that leads to the lowest dof is selected and is the only one that will be further explored.

For our example, we can refer to Fig. 6 to find out that pattern “g” (choices 67.8 and 69.4) has the lowest dof of 65. Step 2 will start with the pattern 1g.

1g			1	4				8		0	48	3	4	0	0	4	5	2	0	3
			4		7	8	6	3	1	1	0	2	2	0	3	0	0	0	0	0
	8		7				9	4		2	12	0	3	0	4	4	5	0	0	2
	4	8	6	7			1	2	3	3	12	0	0	0	0	2	2	0	0	0
	7	1	3	8	2	4	5	6	9	4	7	0	0	0	0	0	0	0	0	0
	9	5	2				8	7	4	5	2	0	0	0	2	3	3	0	0	0
			5		4				8	6	0	3	3	0	4	0	4	2	2	0
	3		9	1	8		4	5		7	0	0	3	0	0	0	2	0	0	3
		4	8			7	3					3	0	0	4	3	0	0	2	2
										43	65									

Fig. 8 Step 2 (1g)

Step 2 (1g) has 24 initial choices, 6 without impact and 4 duplicates, thus 14 different patterns with dof’s ranging from 37 to 59. Pattern 2c offers the lowest dof and will be used for step 3.

2c	5		1	4		9		8		0	57	0	2	0	0	2	0	2	0	2
	2	9	4	5	7	8	6	3	1	1	0	0	0	0	0	0	0	0	0	0
	8		7				9	4	5	2	13	0	2	0	3	3	4	0	0	0
	4	8	6	7	9	5	1	2	3	3	9	0	0	0	0	0	0	0	0	0
	7	1	3	8	2	4	5	6	9	4	2	0	0	0	0	0	0	0	0	0
	9	5	2				8	7	4	5	0	0	0	0	2	3	3	0	0	0
			5		4				8	6	0	2	3	0	4	0	3	2	2	0
	3		9	1	8		4	5		7	0	0	3	0	0	0	2	0	0	3
		4	8		5	7	3					2	0	0	3	0	0	0	2	2
										34	37									

Fig. 9 Step 3 (2c-1g)

The 26 choices of step 3 (2c-1g) lead to 10 different patterns (15 duplicates, one without impact) with dof's ranging from 6 to 28, with "d" having the lowest value.

3d	5		1	4		9	7	8	2	0	75	0	2	0	0	2	0	0	0	0
	2	9	4	5	7	8	6	3	1	1	0	0	0	0	0	0	0	0	0	0
	8		7		1	2	9	4	5	2	6	0	2	0	2	0	0	0	0	0
	4	8	6	7	9	5	1	2	3	3	0	0	0	0	0	0	0	0	0	0
	7	1	3	8	2	4	5	6	9	4	0	0	0	0	0	0	0	0	0	0
	9	5	2			1	8	7	4	5	0	0	0	0	2	2	0	0	0	0
	6	7	5	9	4	3	2	1	8	6	0	0	0	0	0	0	0	0	0	0
	3	2	9	1	8	6	4	5	7	7	0	0	0	0	0	0	0	0	0	0
	1	4	8	2	5	7	3	9	6			0	0	0	0	0	0	0	0	0
										16	6									

Fig. 10 Step 4 (3d-2c-1g)

Step 4 (3d-2c-1g) is the last step along that branch because all of its 12 choices lead to one of two final solutions that are identified in this paper by X and Y.

The "selective" approach is much faster for identifying one solution, 4 steps are required to reach a solution and complete the branch compared to 6 for the first solution and 8 for completing the branch (7a-6a-5a-4a-3a-2a-1a) followed for demonstrating the exhaustive approach.

Difficulty measure of MS-type puzzles

We have seen that SS puzzles have only one solution and that we could imagine extending to them the dof concept as a measure of their difficulty by adding the "step" dof to the initial value. But we cannot use the same approach with MS puzzles because there are different solutions appearing at different places in the choice tree, following different paths and having passed through different stage dof's.

But what would be possible in this direction is to add to the original dof value the sum of all the step dof's required to reach the first solution via a "selective" approach. There is perhaps no guarantee that this technique will get to the minimum total dof value, there might be some unique combination that could lead to a smaller total dof. But this technique properly applied will always yield the same result.

The table in Fig. 11 shows the calculation of the total dof for the above puzzle.

Initial	162	original puzzle
step 1	101	after automatic assignations
step 2	65	after choice made in previous step and automatic assignations
step 3	37	Id.
step 4	6	Id.
Total dof	371	

Fig. 11 Calculation of total dof for a MS-puzzle

To make this measurement stable and reproducible, some considerations must be added for cases of equality at any step (same dof value after next step) and of different solutions with equal total dof.

If two choices lead to the same after-step dof, one should prefer the solution with the larger number of assigned positions; if the equality is not broken, the solution with the smaller number of two's, excluding the eventually of no "two" (one "two" should be preferred to 0 and to 2 or more). If the equality persists, the best would be to work in parallel with both patterns until a final solution is reached and then see which the path is the best in terms of total dof.

Only the final solution with the smallest dof should be retained for defining the difficulty measure. This is does not deal with the "proposed by the author" solution, the subject of the next section.

I have been able to solve another MS puzzle in the "book": #92 seems more "compact" than the 88 in that sense that the first solution (a double one in fact) is obtained at the second step and that branch completely is solved in two more steps with the addition of a third solution. But it can be expected that other branches might be much longer because the large dispersal among the dof values of the potential branching-outs at the first level (8 for the "solution" branch, between 80 and 97 for the other 7 branches). The calculated total dof for #92 is (150 + 109 + 8 =) 267

Spread of difficulty measures for all puzzle types

We can have a good look at the overall distribution of the difficulty measures for 99 of all the puzzles if the "book" (I have had not enough time to solve #100) by adding to the direct puzzle chart (Fig. 3) the new categories Single Step (15, 3 found in Hard, 12 from Devilish but with a complete overlap between them) and Multiple Steps with its only 2 cases.

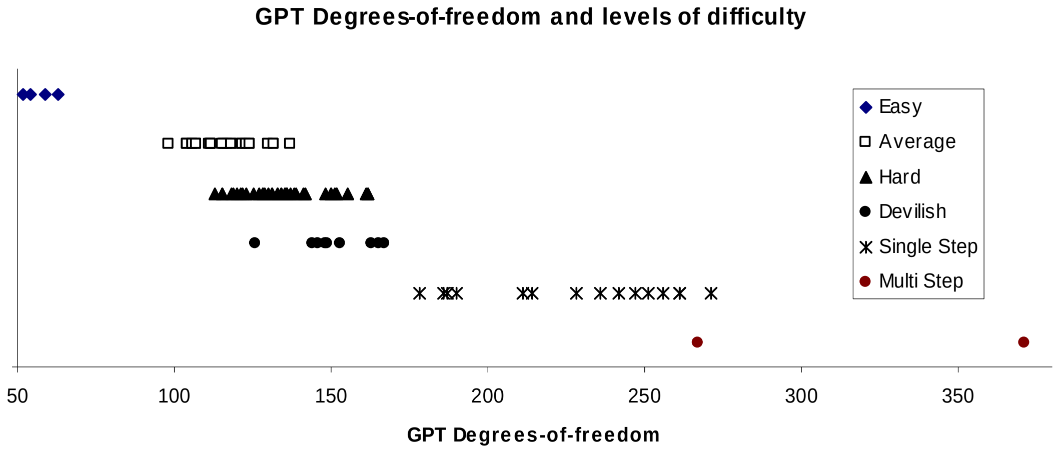


Fig. 12 Global spread of dof's for all types of puzzle in ([1])

This last diagram is the best proof of the difference that can be made between the types of puzzles and the level of difficulty measure by the proposed global dof. It is not surprising to see a small overlap between the highest SS and the smaller MS – and that overall will become more pronounced as the size of the measurements sample increases – because "compact" MS puzzles may be easier to solve than SS puzzles where very few automatic assignations can be made initially.

Recommendations for the use of a difficulty measure

I recommend using for measuring the difficulty of Su Doku puzzles the dof concept associated with the type distinctions as follows

D-type Puzzles that can be solved deterministically	Total dof of Initial Global Performance Table
SS-type Puzzles that can be solved with a single choice after automatic assignation phase	Sum of "Total dof of Initial Global Performance Table" + "Total dof of Global Performance table after automatic assignations"
MS-type Puzzles that require more than one choice in sequence to be solved	As above plus minimum dof among the alternative choices possible at each step, that choice being the start of a new step.

I would not make any immediate recommendation about the use of a scale based on this measure. A scale is a very subjective construct and the information that is passed to the user most relative. If an "author" was to indicate how he defines "his" scale (let us say by giving the breakpoints in dof for each level), the user would be able to attach "labels" to some objective reference measure, the dof, and perhaps relate that to his degree of experience in solving puzzles.

Recommendations for the disclosure of a solution

The disclosure of a solution is a widespread practice, be it at the end of a book or in the next “issue”. It is generally viewed as a good practice giving the solver a chance to make sure there was a solution and that he found the right one. But it can be defeating its own purpose if the proposed puzzle offers more than one solution. If the user has found a different answer, has he failed?

We have seen that D-puzzles have mechanistically a unique solution, that SS-puzzles have also most probably a unique solution[2] (however one can imagine that some specific puzzle could offer solutions in part “symmetrically” different on the cells not automatically assigned), but that MS-puzzles offer several distinct solutions. Hence these general rules for disclosure:

D-type	The unique solution
SS-type	The unique solution, but in case of symmetrical differences, both solutions
MS-type	The “selective” solution obtained along the shortest path giving the minimum total dof, plus a mention that it is only one of many possible solutions. All solutions in case of ties.

[2] I cannot ascertain the uniqueness of a SS-puzzle solution but from the deduction I made earlier. I have found out that these kinds of statements have a tendency to be shot down at the first occasion. I must then temperate this assertion.

Annex A

Template to help solving Su Doku problems

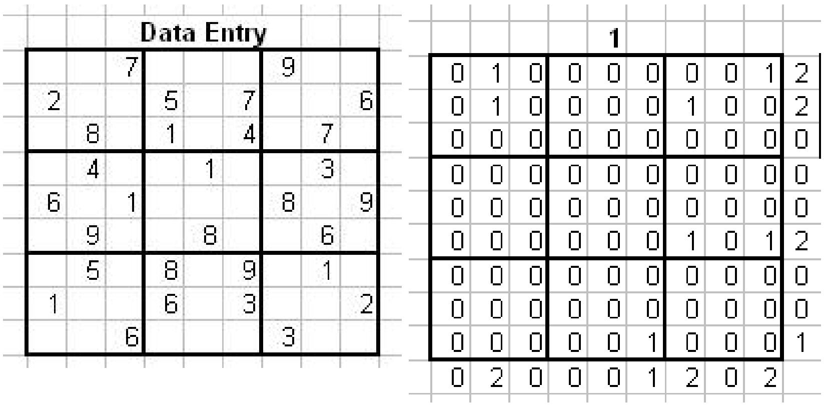
When I began exploring the subject, I just wanted to systematize Su Duko rules in order to better understand ways to solve Su Doku problems. I ended up with a template (an Excel worksheet) that makes them operational in the limited context of support to solution building. I never intended to create a program that will solve problems by itself; this template only registers the choices made by the user and simply indicates potential choices that are left.

We review first the basic rules and their implementation in a worksheet. Then we look into the various parts of the template to learn about their roles before we establish the procedures useful for supporting a search for solution.

The XLS file can be freely downloaded as a .ZIP file from the author's site (sudoku_template.zip) where more documentation on SuDoku puzzle solving is available.

Basic rules and implementation

Let us review first the basic rules. Su Duko uses integers from 1 to 9 and a 9*9 square grid. Starting with a partially filled grid, one must complete it in such a way that each integer is present once and only one time in each row, in each column and in each of the 3*3 groups of 3*3 cells, subsets of the main grid.



Data Entry

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

1

0	1	0	0	0	0	0	0	1	2
0	1	0	0	0	0	1	0	0	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	1	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	1
0	2	0	0	0	1	2	0	2	

Example of a "problem" and "performance" for integer 1

The template is based on the concept of "Integer performance table" that is a 9*9 grid initially filled with 1's. There is a similar table for each of the 9 integers. In such tables, 1 means that the cell is open to receive that integer, 0 that the integer cannot be assigned to that position. The change from 1 to 0 is made in several circumstances; one most obvious and not explicitly stated in the rules is that the position is already taken by another integer (initially the cells defined in the "problem"). Other circumstances are the direct translations of the 3 rules: if the integer is present in a cell, all the cells from that row, all from that column and all from the square group are set to 0.

The double illustration above shows how a problem is translated into an "Integer performance table” for integer 1. Five of the 9 square 3*3 groups are filled with 0 as the integer 1 is present 5 times in the problem and each time in a different group (one of the rules); furthermore after the "zeroing" of the 5 different rows and the five different columns in which 1 is present (the other rules) and of the cells containing the other integers given in the problem, there remains only 7 cells into which the integer 1 could be placed (only 4 are required, 5 being originally assigned).

Totals for rows and columns have been added. In this example, totals for row 9 (bottom) and column 6 (from the left) are equal to 1. That means that as the integer 1 must be present in that row and in that column, it MUST BE PLACED IN THAT POSITION (96). The same conclusion would have been reached after noting that the total for the square group (32) (bottom, center) is of 1; the only place for integer 1 in that group is position (96). If that example shows some concordance between the different approaches (row, column and group totals), it is not by all means a general rule.

One must be aware thus that the number of ones in the various totals does not represent the number of integers that must be assigned because the same potential position can be detected by a one in any total or in any 2 or even in the 3 totals. The ones play only the role of locators, not of different solutions

When starting to interpret the results from the implementation of the rules, we have slipped from dealing with a simple recording of the situation to an elementary strategy of search for solution. A short step further will place us in the midst of problem solving. It is taken by creating another table named "Global performance" in which each cell is the sum of the cells in the same position in all 9 integer performance tables. The numbers making up that global performance table will vary from 0 (the cell is already filled and no other integer can be assigned to it) to 9 (any integer from 1 to 9 can be assigned to that cell; that situation is purely theoretical; I have never encountered values higher than 7). In such as table, 1 means that there is only one integer that could be placed there (it must be identified by inspecting the integer performance tables and finding for which integer the position corresponding contains a 1) and that integer MUST BE PLACED IN THAT POSITION.

The proposed procedures for using the template are based on these simple but constraining findings.

Template overview

NB. 1 - The only zones accessible to the user for data entry are the" Work Area" and "Data Entry" tables

2 - There are hidden cells in rows 50 to 150 with intermediate results not useful to the user.

Explanation of the tables:

- Work Area

The solution will be built in that table, adding new digits one at the time after copying the “problem” from the Data Entry table..

- Global Performance

The displayed numbers are the sums of the corresponding cells in the nine Integer Performance Tables. They vary from 0 to a theoretical 9.

- Impossibility Warning

XXX is displayed in a cell with a global performance valued 0 and in which no digit has been assigned. In such case, it would be impossible to complete the grid.

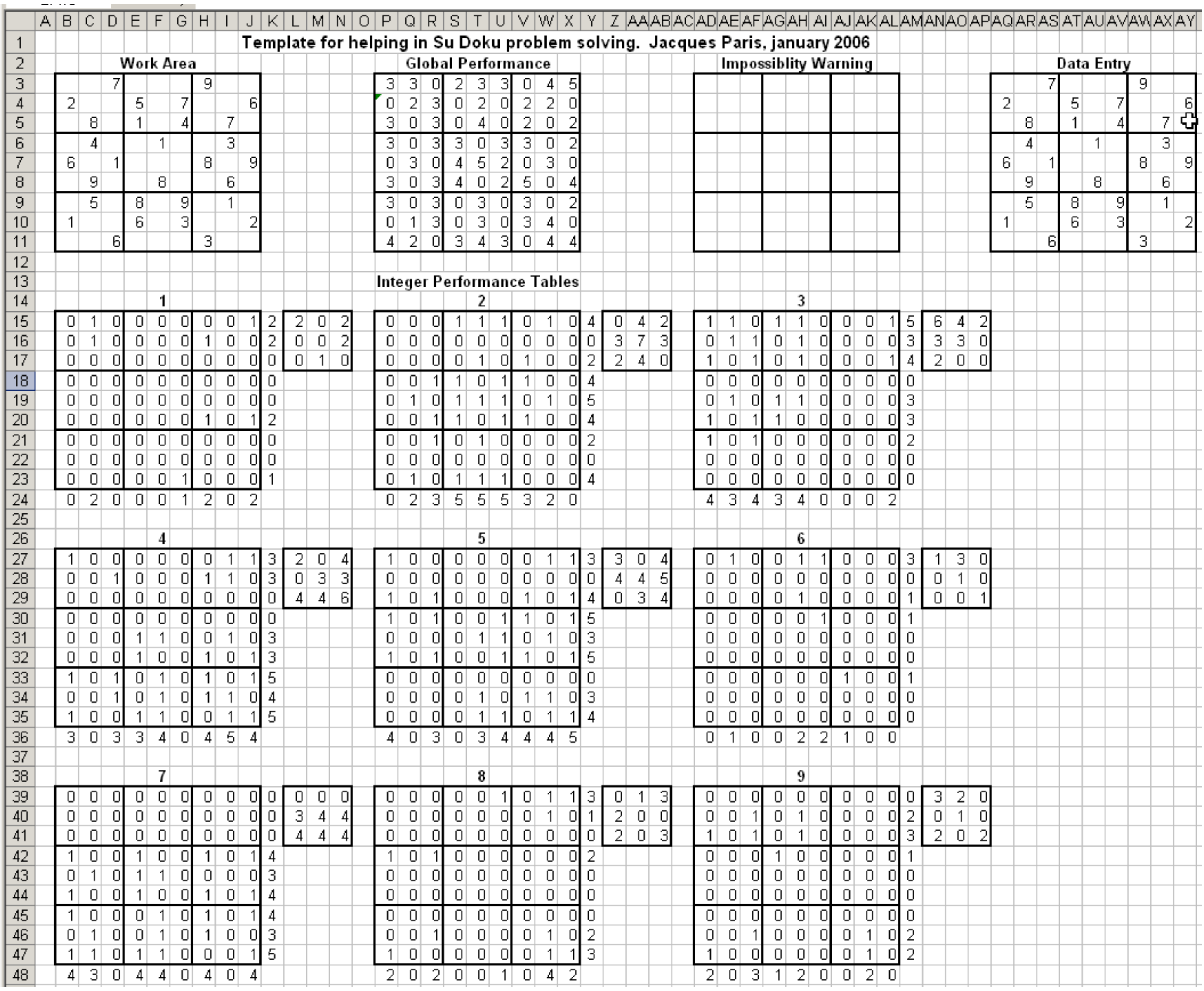
- Data Entry

The Su Doku "problem" is transcribed into that area

- Integer Performance Tables.

There is an area for each digit 1 to 9. It is composed of a 9*9 individual cell performance table, a column of row totals, a row of column totals and a small 3*3 table containing the sums of the square groups..

An individual cell contains 0 if the integer cannot be placed in it (e.g. another integer is already assigned to it) or 1 if it can be assigned to the it The sums of rows, columns and groups are the simple count of the ones present in the corresponding element.



Template for helping in Su Doku problem solving. Jacques Paris, january 2006

Work Area

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

Global Performance

3	3	0	2	3	3	0	4	5
0	2	3	0	2	0	2	2	0
3	0	3	0	4	0	2	0	2
3	0	3	3	0	3	3	0	2
0	3	0	4	5	2	0	3	0
3	0	3	4	0	2	5	0	4
3	0	3	0	3	0	3	0	2
0	1	3	0	3	0	3	4	0
4	2	0	3	4	3	0	4	4

Impossiblity Warning

Data Entry

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

Integer Performance Tables

1

0	1	0	0	0	0	0	0	1	2	2	0	2
0	1	0	0	0	0	1	0	0	2	0	0	2
0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	1	0	1	2			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	1	0	0	0	1			
0	2	0	0	0	1	2	0	2				

2

0	0	0	1	1	1	0	1	0	4	0	4	2
0	0	0	0	0	0	0	0	0	0	3	7	3
0	0	0	0	1	0	1	0	0	2	2	4	0
0	0	1	1	0	1	1	0	0	4			
0	1	0	1	1	1	0	1	0	5			
0	0	1	1	0	1	1	0	0	4			
0	0	1	0	1	0	0	0	0	2			
0	0	0	0	0	0	0	0	0	0			
0	1	0	1	1	1	0	0	0	4			
0	2	3	5	5	5	3	2	0				

3

1	1	0	1	1	0	0	0	1	5	6	4	2
0	1	1	0	1	0	0	0	0	3	3	3	0
1	0	1	0	1	0	0	0	1	4	2	0	0
0	0	0	0	0	0	0	0	0	0			
0	1	0	1	1	0	0	0	0	3			
1	0	1	1	0	0	0	0	0	3			
1	0	1	0	0	0	0	0	0	2			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
4	3	4	3	4	0	0	0	2				

4

1	0	0	0	0	0	0	1	1	3	2	0	4
0	0	1	0	0	0	1	1	0	3	0	3	3
0	0	0	0	0	0	0	0	0	0	4	4	6
0	0	0	0	0	0	0	0	0	0			
0	0	0	1	1	0	0	1	0	3			
0	0	0	1	0	0	1	0	1	3			
1	0	1	0	1	0	1	0	1	5			
0	0	1	0	1	0	1	1	0	4			
1	0	0	1	1	0	0	1	1	5			
3	0	3	3	4	0	4	5	4				

5

1	0	0	0	0	0	0	1	1	3	3	0	4
0	0	0	0	0	0	0	0	0	0	4	4	5
1	0	1	0	0	0	1	0	1	4	0	3	4
1	0	1	0	0	1	1	0	1	5			
0	0	0	0	1	1	0	1	0	3			
1	0	1	0	0	1	1	0	1	5			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	1	0	1	1	0	3			
0	0	0	0	1	1	0	1	1	4			
4	0	3	0	3	4	4	4	5				

6

0	1	0	0	1	1	0	0	0	3	1	3	0
0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	0	0	1
0	0	0	0	0	1	0	0	0	1			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	1	0	0	1			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	1	0	0	2	2	1	0	0				

7

0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	3	4	4
0	0	0	0	0	0	0	0	0	0	4	4	4
1	0	0	1	0	0	1	0	1	4			
0	1	0	1	1	0	0	0	0	3			
1	0	0	1	0	0	1	0	1	4			
1	0	0	0	1	0	1	0	1	4			
0	1	0	0	1	0	1	0	0	3			
1	1	0	1	1	0	0	0	1	5			
4	3	0	4	4	0	4	0	4				

8

0	0	0	0	0	1	0	1	1	3	0	1	3
0	0	0	0	0	0	0	1	0	1	2	0	0
0	0	0	0	0	0	0	0	0	0	2	0	3
1	0	1	0	0	0	0	0	0	2			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	0	1	0	0	0	0	1	0	2			
1	0	0	0	0	0	0	1	1	3			
2	0	2	0	0	1	0	4	2				

9

0	0	0	0	0	0	0	0	0	0	3	2	0
0	0	1	0	1	0	0	0	0	2	0	1	0
1	0	1	0	1	0	0	0	0	3	2	0	2
0	0	0	1	0	0	0	0	0	1			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	0	0	0			
0	0	1	0	0	0	0	1	0	2			
1	0	0	0	0	0	0	1	0	2			
2	0	3	1	2	0	0	2	0				

Procedures

1 - First, erase all entries from the "Data entry" table. Then, enter numbers as given in the problem.

2 - Copy "Data entry" table over "Work area". All new entries will be made exclusively in that area until the table is filled up and the problem solved.

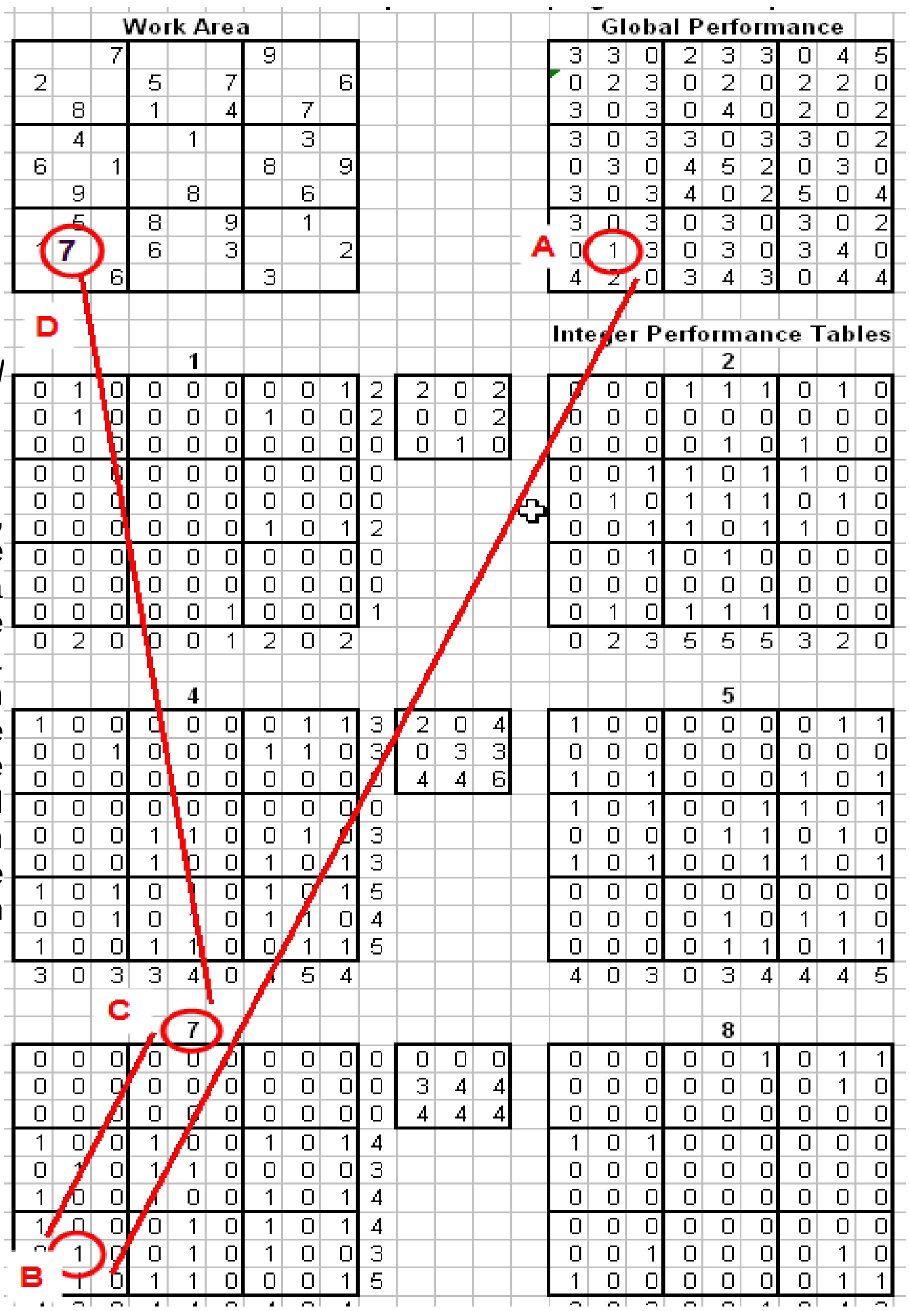
3 - To assign a number to a cell, look for a 1 in the following areas and proceed as stated

NB The order in which numbers are assigned on the basis of these "1" is totally irrelevant as long of course there are 1.

a

- *in the global performance 9*9 table*.

From that table, record the position (A) of a 1, find the integer with a 1 in that position (B), record the value of the integer (C) and enter it (D) in the same position as in (A).



Work Area

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

Global Performance

3	3	0	2	3	3	0	4	5
0	2	3	0	2	0	2	2	0
3	0	3	0	4	0	2	0	2
3	0	3	3	0	3	3	0	2
0	3	0	4	5	2	0	3	0
3	0	3	4	0	2	5	0	4
3	0	3	0	3	0	3	0	2
0	1	3	0	3	0	3	4	0
4	2	0	3	4	3	0	4	4

Integer Performance Tables

1

0	1	0	0	0	0	0	0	1	2
0	1	0	0	0	0	1	0	0	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	1	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	1
0	2	0	0	0	1	2	0	2	

2	0	2
0	0	2
0	1	0

2

0	0	0	1	1	1	0	1	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	1	0	0
0	0	1	1	0	1	1	0	0
0	1	0	1	1	1	0	1	0
0	0	1	1	0	1	1	0	0
0	0	1	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	1	0	1	1	1	0	0	0
0	2	3	5	5	5	3	2	0

4

1	0	0	0	0	0	0	1	1	3
0	0	1	0	0	0	1	1	0	3
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	1	0	3
0	0	0	1	0	0	1	0	1	3
1	0	1	0	1	0	1	0	1	5
0	0	1	0	1	0	1	1	0	4
1	0	0	1	1	0	0	1	1	5
3	0	3	3	4	0	4	5	4	

2	0	4
0	3	3
4	4	6

5

1	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	1	0	1
1	0	1	0	0	1	1	0	1
0	0	0	0	1	1	0	1	0
1	0	1	0	0	1	1	0	1
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	1	1	0
0	0	0	0	1	1	0	1	1
4	0	3	0	3	4	4	4	5

7

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0	1	4
0	1	0	1	1	0	0	0	0	3
1	0	0	1	0	0	1	0	1	4
1	0	0	0	1	0	1	0	1	4
0	1	0	0	1	0	1	0	0	3
[illegible]	1	0	1	1	0	0	0	1	5

0	0	0
3	4	4
4	4	4

8

0	0	0	0	0	1	0	1	1
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	1	0
1	0	0	0	0	0	0	1	1

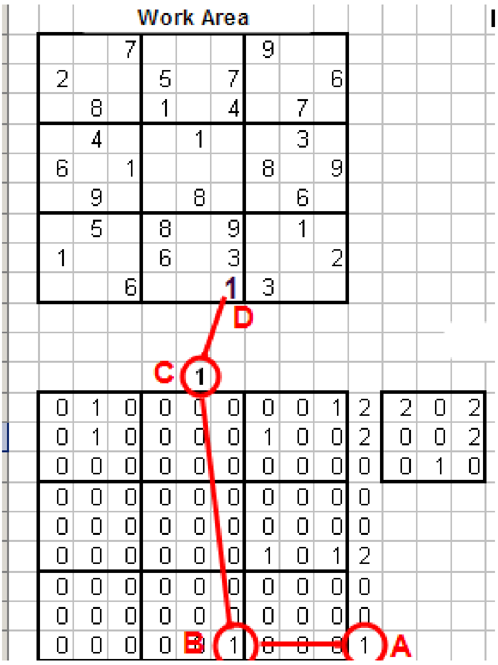
b

- *or in the row totals*.

The integer with 1 in its row total (A) will be assigned (C, D) at the same position as in (B).

- *or in the column totals*.

(Transpose the situation from previous case). The integer with 1 in its column total (A) will be assigned (C, D) at the same position as in (B).



Work Area

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

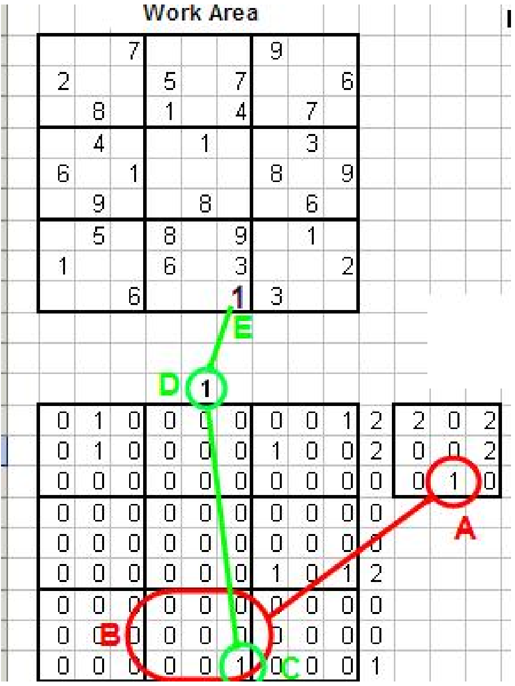
0	1	0	0	0	0	0	0	1	2
0	1	0	0	0	0	1	0	0	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	1	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	1

2	0	2
0	0	2
0	1	0

c

-or in the square group totals.

When a group total is = 1 (A), the cell position must be found in the Integer Performance Table (B,C) in order to assign that integer (D) in the same position (E).



Work Area

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

0	1	0	0	0	0	0	0	1	2
0	1	0	0	0	0	1	0	0	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	1	2
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	1

2	0	2
0	0	2
0	1	0

-

When the global performance table contains only 0, the problem is solved.

4 - If the Global Performance table contains only values >1, and if there are no 1 in any of the row, column or group totals, the search for a solution requires the

user to make a choice. He must choose between the possibilities that are left to find the one (and only one) that will work. If, for example, there is a 2 in one of the cells, it means that two integers can occupy that space; the choice will have to be between these two values that can be identified by the inspection of the Integer Performance tables.

Before proceeding further, it is highly recommended to **make a copy of the template in its present state on a different worksheet (or in the Data Entry area)**; if the choice proves to be the wrong one, there will be no need to start the procedure again from scratch, the previous acceptable assignments being saved.

One of the possibilities must be chosen, then the procedure described in 3 can continue.. There are 3 possible outcomes:

> 1 - integers can be assigned to all the remaining cells of the table without any maker appearing in the "Impossibility Warning" table. The problem is solved.
>
> 2- a maker appears in the "Impossibility Warning" table. That means that the integer chosen to start that phase was not the good one. One must backtrack and restart with the alternative (hence the usefulness of the copy made previously). If the choice was between 2 possibilities, then the second one will be the good one. But if there were more than 2 possibilities, the second choice may turn out to be good or bad (hence the recommendation to choose a cell where there is a 2, rather then a 3)
>
> 3 - a sequence of cells can be filled up without any maker appearing in the "Impossibility Warning" table but without completing the grid because there is no more 1. One must start another round as described in 4 -.

NB As soon as a "phase 4" is started, it is vital to check after each assignment **if any marker appears in the "Impossibility Warning" table**. When one appears, **one must stops immediately and re-start with an alternative choice.**

Impossibility markers can appear also for purely technical reasons, for example if an integer has been assigned to the wrong place, if there was a transcription error of the problem to the Data Entry table or for any mistake. The template is not designed to pickup such errors of which the consequences (and the impossibility markers) may not appear immediately. Before deciding that a detected impossibility must be taken for real, it is recommended to redo the sequence from the choice that ended up in the apparition of the markers to eliminate any handling error.

Annex B

On the way to “hard” puzzles

I wanted to see if it was possible to make an “easy” puzzle harder and what would happen on the way to “hardness”. I chose a very easy puzzle (#4 in the ‘Book”, 36 clues, dof of 52) and devised a procedure to eliminate clues. I wanted to make that procedure independent of my perception of what was happening to the puzzle, and easy to use. I started with one that is by no means unique and I can imagine several alternatives[3] that could be more intelligent and productive but some of the results I obtained along the way were worth reporting.

	8	5				2	1		36
	9	4		1	2			3	10
			3			7		4	22
5		3	4		9				9
	4		2		6		3		4
			1		3	9		7	0
6		8			5				0
1			8	4		3	6		0
	2	7				8	9		
									52

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

Fig. B1 The original #4 puzzle and its “direct” solution

I decided to eliminate clues in the increasing order of dof calculated after pulling that clue from the puzzle, leaving all the other in place. This is not a very logical way to proceed because the consequence of removing two clues cannot be predicted from the impact of removing them one at the time, the other remaining in the puzzle. But, as I mentioned, I was more interested in the general results than in detailed information.

[3] A stricter way would be to select the clue that once removed would yield the smallest dof at each step (and not on the original puzzle as I did here); or the highest dof. One could also rely on randomized choices. See at the end of this annex :A random approach with unexpected results”

I was able to remove 8 clues without any significant change[4] then with the next one (84.8) the puzzle became an SS-type.

	8	5				2			27
	9	4		1				3	4
			3			7		4	16
5					9				17
	4		2		6				11
			1			9		7	6
6					5				0
1				4			6		0
	2	7				8	9		
									107

3	8	5	7	6	4	2	1	9	77
7	9	4	5	1	2	6	8	3	0
2	1	6	3	9	8	7	5	4	4
5	7	3	4	8	9	1	2	6	0
9	4	1	2	7	6	5	3	8	0
8	6	2	1	5	3	9	4	7	0
6	3			2	5	4	7	1	0
1	5			4	7	3	6	2	0
4	2	7	6	3	1	8	9	5	
									4

Fig. B2 After removal of 8 clues

This puzzle has a very low original dof for a stepped puzzle and its calculated value at the end of the automatic assignations is also very low. The combined value of 117 (calculated as I recommend it) is even more at the extreme low end for such puzzle. But one must concede that the “step” has not increased the difficulty by much: two integers (8 and 9) can fill those 4 cells and when you choose one for a cell, it is not too complicated to fill the other 3. But that creates an interesting “solution” for this single step puzzle.

73.8

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

73.9

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

Fig. B3 Dual symmetrical solutions.

I suggested in the main text that SS-type should have a unique solution, but on an intuition, I opened the door to possible “symmetrical” solutions, and here is one in all its splendour.

One more clue is removed (29.3), and the puzzle becomes an MS-type.

[4] The sequence of removals was : 58.3, 26.2, 66.3, 87.3, 73.8, 18.1, 33.3, 34.4 (34.4 means row 3, column 4, integer 4)

	8	5				2			25
	9	4		1					2
			3			7		4	11
5					9				19
	4		2		6				16
						9		7	7
6					5				1
1				4			6		0
	2	7				8	9		
									130

3	8	5	7	6	4	2	1	9	73
7	9	4	5	1	2	6			0
2	1	6	3	9	8	7	5	4	8
5	7	3	4	8	9	1	2	6	0
9	4	1	2	7	6	5			0
8	6	2	1	5	3	9	4	7	0
6	3			2	5	4	7	1	0
1	5			4	7	3	6	2	0
4	2	7	6	3	1	8	9	5	
									8

Fig. B4 After removal of a ninth clue

Another block of 4 cells (upper right, split in two in fact) accepts the integers 3 and 8 for a “symmetrical” solution. If we solve that new block as a first step, we have get 2 distinct matrices and each one leads to the same situation as with the SS-type. This MS-puzzle thus yields 4 solutions and has a combined dof of (130 + 8 + 4 =) 142. It is extremely low, but it is not a complicated MS-puzzle, its solutions are easy.

A random approach with unexpected results.

I did not resist long to the possibilities offered by a random approach to clues removal. I devised a small tool to generate a list of clues to be removed in sequence with a constant random function applied to the remaining clues. A prepared a list of 12 clues but I did not have to use more than 3 of them before generating a Stepped puzzle. What did happen?

The two areas we just identified that generate symmetrical solutions, contain diagonally one two 8, the other a single 3. It seems that this puzzle does not generate other constraints to define the integers to put in the other diagonal than those due to the presence of these 3 and 8. If these clues are removed (the one 3 or the two 8), the corresponding area becomes not determinate enough, hence the multiple solutions from which one must choose in this new “step”.

The unexpected result of that experience is the confirmation that clues contain information (constraints resulting from the application of basic rules) that is “spatially strategic”, i.e. clue position can have impacts beyond those possibly measured in a static situation, impacts that can only be detected in the process of solving the puzzle.

Annex C

A dynamic measure of “ease”

The proposed DOF calculation on the GPT as an indicator of difficulty revealed by the number of potential choices has several limitations; one in particular is that it deals only with the initial puzzle and not with the process for solving it. It does not reflect the personal experience of solving a puzzle with its continuous search for the next move.

The template I designed gave me the idea to use the number of choices for assignation present (thee “must” assign cells) at any time to measure the instantaneous “ease” for finding the next move. I made the assumption that a puzzle solver will find it easier if more possibilities are offered than if those are few; locating a possibility when many exists is certainly easier than finding the single one lost somewhere in the puzzle..

Assination choices are revealed in the template by 1 in the “totals” areas for rows, columns and groups. I chose to use the total of all these ones as the indicator of ease, the higher that number, the easier finding the next move.

One may argue that the real choices are the 1’s in the Integer Performance Tables corresponding to a 1 in one of the “totals” areas, i.e. the choice is the location to which the integer must be assigned; as the same location can be detected by any combination of the 3 types of “totals” and thus contribute to the proposed measure by 1, 2 or 3 points, the proposed measure does not indicate the real number of choices.

This argument cannot be dismissed and forces me to reformulate my definition by saying that the aim is to measure the available information directly related to the real choices. This is an even better approximation of solver behaviour; one would reach a real choice (the position) with the help of some indicators, the “totals”; the more indicators the easier the choice.

I have voluntarily not included the GPT ones in the total indicator. GPT is not available as such to the solver who does not use the template. GPT would enter into play when there are no ones left in the totals, the total indicator being 0 in that case; the user will have to use the GPT to launch the solving process again with an assignation defined from its information. That re-launching is always possible in the case of D-puzzles if not it would be an S-type puzzle and I do not want to deal with SS and MS puzzles yet as this is only an exploratory discourse.

As a first experiment, I have recorded the “totals” measure before each automatic assignation for 4 D-puzzles that could be ranked from easy to devilish, and combined the 4 line plots in diagram (Fig. C1)

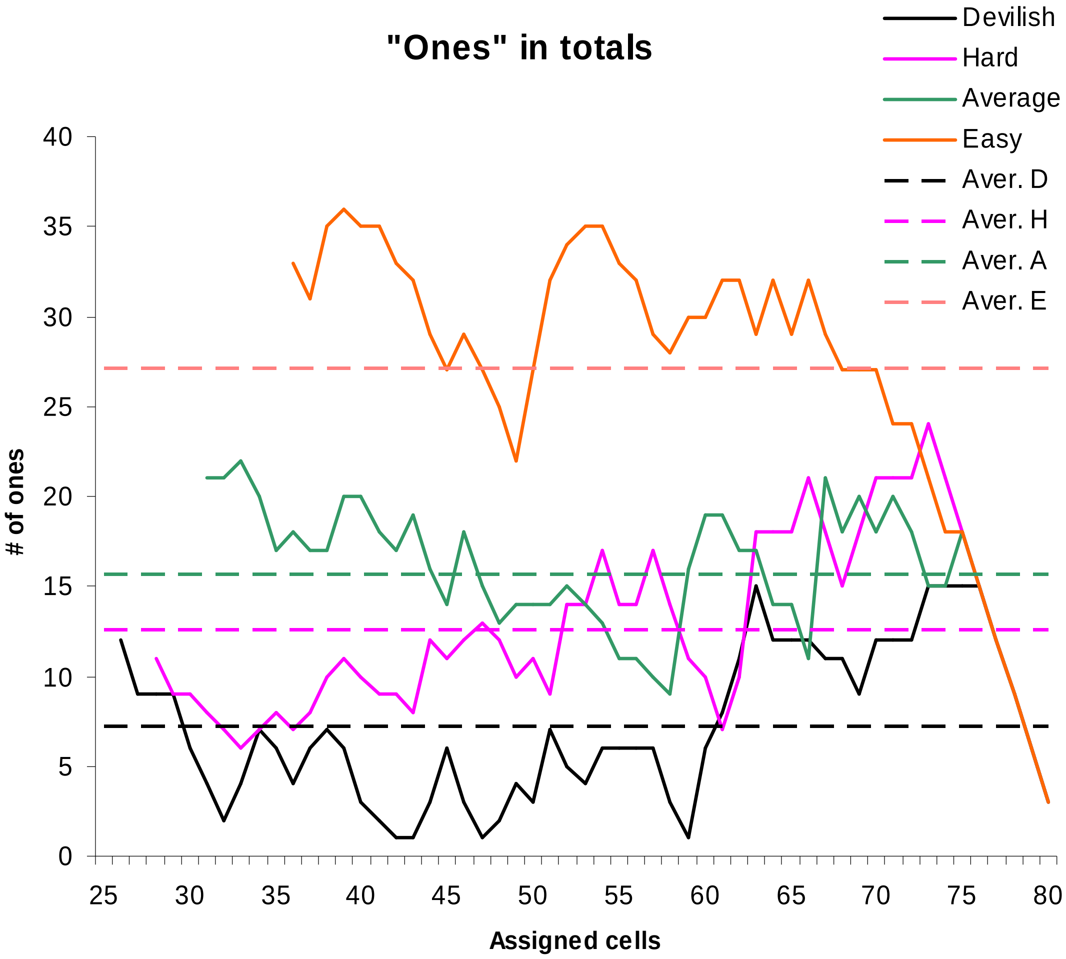


Fig. C1 “Totals” indicators for 4 D-puzzles

This diagram yields many indications that would have to be validated or dismissed after more extensive tests

1 – The different “curves” do not start from the same point because the puzzles have a different number of clues (assignations already done). In this sample, the higher the curve (level of ease) the larger the number of clues; but it may not always be the case if we take into account what we found between number of clues and “declared” level of difficulty (see Fig. 2).

2 – Fluctuations between consecutive assignations are not very pertinent in themselves because they may differ if the order of assignations is changed; most of the time there are several alternatives for an assignation and no prescribed

rules to choose among them. One should rather take into account the overall look of the “curve”.

3 – If the 4 curves are relatively distinct in the area below roughly 60 assigned cells, their overlapping increases when getting closer to the end. That overlapping seems to be into two parts, first a narrowing of the range of values around the 70 cells, then a common drop to the end. I would explain that behaviour by the decease in the number of non-assigned cells, the remaining information (the constraints applied by the assigned cells) exerting an increase pressure on a progressively diminishing number of cells. The final drop is a series of values reduced by 3 at each assignation and starting from a high of 24 to a low of 15. That these values are multiples of 3 can be explained by the way the indicator is built, i.e. summing up 3 “totals” in which each assignation seems to be present due to the “concentration” of the constraints on few cells.

4 – Averages computed on all the assignations differentiate clearly the 4 puzzles as far as level of ease is concerned.

5 – We can detect interesting variations in the overall shape of the curves if we use the average lines as reference. For the 3 curves excluding the easiest one, the first portion is globally bellow the average, and above it in the second part corresponding to the “concentration” noted in point 3- until the “last drop”.

Those observations give us some leads for further investigations such as

A – Should the “final drop” be eliminated from the calculation? If it is the same for all, its inclusion could blur the indicator.

B – Should the measure be calibrated by the number of remaining “free” cells? That “average per free cell” indicator will automatically accentuate the observed increase in the portion of the curves before the last drop. My first reaction would be to reject that idea on the basis that the solver is probably more sensitive to the globally perceivable numbers than to an abstract measure such as the average, but there might be some intrinsic value to it.

C – Is there a way to make the average more representative of the overall curve when there are such distinct levels between the first (beginning, below average) and second part (“concentration”, above average)?

D – Finally, how could this measure be applied to Stepped-puzzles to reflect the diminution of “ease” other than by recording a simple 0 for the “step” assignation?

We cannot close that exploration without comparing different measures: the “average ease” just devised and the “degrees of freedom” of the GPT

label	dof	aver. ease
Easy	35	27.1
Average	98	15.7
Hard	119	12.6
Devilish	142	7.2

Fig. C2 Measures for 4 D-puzzles

There is no doubt that on such a small sample the inverse relation between “dof” and “average ease” is well established.

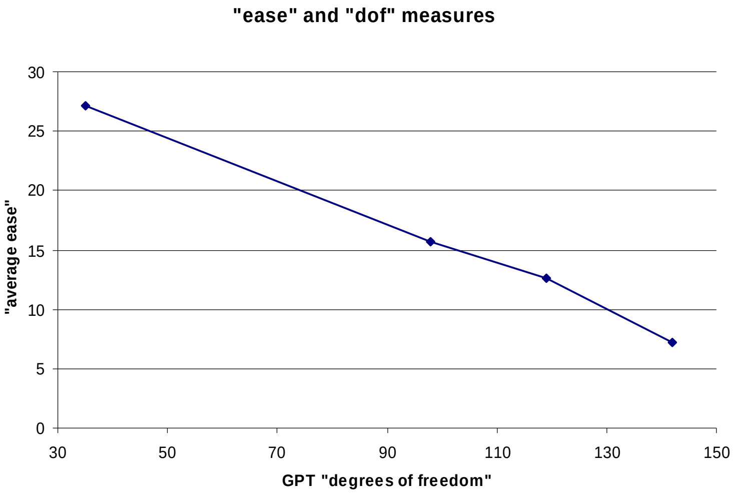


Fig. C3 Relationship between “ease” and “dof” measures

If that relationship is as strong as it seems, what would be the advantage to adopt a measure requiring a calculation to be performed over the entire solving process and that may not in the end give more information for scaling that one computed in one shot without having to enter the solving process?