数独の解生成と解に対する番号付け

戸神星也
理学部 情報科学科　指導教官　渡辺治

概要

パズルの一種に「数独」というものがある.「数独」というのは図 1 のような盤面が与えられたときに，ルール (後述) にしたがって解を導くパズルである．例えば，この盤面に対しては図 2 が解となる．

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

図 1: 数独の例

本研究では数独の解盤面 (図 2 は解盤面の一つである) に注目し，解盤面に対する番号付けを提案する．また，番号付けを効率よく行う方法を既存の研究を元に考え出し，更にその改良方法について提案する．

1　数独の解盤面と番号付け

数独とは図 1 のようなパズルであり，既に埋まっているマスの数字を頼りに空いているマスを埋めていくパズルである．この際，一種の魔方陣を完成させるように数字を埋めていく．以下，正確なルールを記す.

- 空いている**マス**に 1 から 9 までどれかの数字を埋める
- 同じ**列**，同じ**行**，同じ**ブロック** (太線で囲まれた 3×3 のマスの集合) に 1 から 9 の数字をそれぞれ 1 つずつ埋める

数独の問題としては図 1 が与えられ，図 2 のような解を導く．また，図 2 のように導かれた解を**解盤面**と呼ぶことにする.

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

図 2: 解盤面の例

この解盤面の場合の数は Jarvis ら [1] によって計算されていて，6670903752021072936960 通りであることが分かっている.

本研究ではこれら解盤面に対する番号付けを試みる．即ち，解盤面の場合の数を N_0 としたときに，それぞれの解盤面に対して 1 から N_0 までの番号を付けることを考える．そしてこのとき，任意の解盤面からそれに対応する番号を出力する方法，及び任意の番号から解盤面を出力する方法を考える．これらを実現するために

は，バックトラック法などにより (コンピュータを用いて) 解盤面を列挙していくことで実現可能ではあるが，単純な方法では実行時間がかかりすぎる．本研究のテーマは，この実行時間をいかに短くしていくかということである．

2　番号付けの実現

解盤面の作成は今のところ探索により行う方法しか知られていない．よって，番号付けの実行時間を短くするためには，探索する範囲を狭くしていく方法がまず考えられる．その為に解盤面の**標準形**というものを定義した．図 3 において $a_1 < a_2 < a_3$, $a_4 < a_5 < a_6$, $a_1 < a_4$, $b_1 < b_2 < b_3$, $b_4 < b_5 < b_6$, $b_1 < b_4$ となっているような形の解盤面を標準形の解盤面といい，任意の解盤面は標準形に変換可能である (変換方法がごく簡単に記述でき，逆変換も容易であるという意味)．標準形の解盤面は解盤面全体に対して 1.8×10^9 分の 1 程度しか存在しないため，任意の数独を標準形に変換 (**標準化**) することにより解盤面の探索範囲を狭めることができる．

1	2	3	a_1	a_2	a_3	a_4	a_5	a_6
4	5	6						
7	8	9						
b_1								
b_2								
b_3								
b_4								
b_5								
b_6								

図 3: 標準形の数独

また，上 3 行のみを考えたとき，図 4(上) を含む解盤面の総数と図 4(下) を含む解盤面の総数は同じである．4 行目以降のマスに対する制約条件が一切変わらないためで，このような変換は記憶領域の削減 (後述) に有用である．

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

各列について
丸数字部分を入れ替え

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

図 4: その他の変換

3　研究の成果

標準化を利用した方法により，解盤面の番号付けは現実的な速度 (一般的な家庭用 PC で平均 5 分程度) で実行可能となった．

更なる改良として，探索の結果現れる解盤面を全てデータとして保存しておくという方法を実践した．単純に全ての解盤面を保存するとなると膨大な記憶領域が必要であるが標準化及び図 4 のような変換を利用することにより 40GB 程度 (圧縮すれば 4GB 程度) に抑えることに成功した．図 4 のような変換は Jarvis らによって 7 つ提案されているが，本研究ではこれら 7 つの変換に含まれない変換を発見し，記憶領域の削減に役立てている．この方法を用いると実行時間は更に短くなり現在は一般的家庭用 PC において平均 10 秒程度である．現在，実際にこのプログラムを http://doorgod.org/sudoku/ で公開している．興味のある方は参照してもらいたい．

参考文献

[1] Bertram Felgenhauer and Frazer Jarvis. Sudoku enumeration problems. http://www.afjarvis.staff.shef.ac.uk/sudoku/, 2006