

研究レポート

研究タイトル「5次魔方陣を求める」

(継続研究)

学校名「広島大学附属福山中・高等学校」

研究者名「田中愛登」

研究期間「2014年4月から2015年9月」

5次魔方陣を求める

広島大学附属福山高等学校1年 田中愛登

○研究目的

魔方陣とは縦・横・斜めのどの列についても、列の数字の総和が等しくなる様に数を並べた数の表で、回転・対象移動して得られるものを同一視すると五次魔法陣は2億7530万5224個存在する。

私は昨年、高校生が筑波大学のスーパーコンピューターを使い五次魔方陣の全解を求め、その計算時間が最短で約2時間36分であるという話を聞き、高速化の余地があるのではないかと考えた。そこで昨年度は、学校の情報処理演習室のコンピューター44台(176個のコア)と新しいアプローチのアルゴリズムを使い、より短い時間で計算することを目標とした実験を行い、約44秒で求める事に成功した。このアルゴリズムは、筑波大学での先行研究と比較して魔方陣となる候補をより効率的に絞り込むと同時に魔方陣となり得ない場合を計算領域から除く事によって高速化をめざした。この昨年度の取り組みは広島大学理学部での中高校生科学シンポジウムで発表した。本年度の取り組みではさらにアルゴリズムを改良することによって30秒以下で求める事を目指し実験を行った。

○研究手法

- ・ソフトウェア (プログラム)

今回の計算プログラムにはプログラミングと管理のしやすさから Python を使用した。Python はインタプリタ言語であるので高速化のために `shedskin` によって C++ 言語から `exe` に変換して実行した。また、昨年度は全解の個数だけを入力するプログラムと全解の個数とそれぞれの魔方陣を入力するプログラムの二種類を作成したが、今年度は全解の個数とそれぞれの魔方陣を入力するプログラムのみを作成した。

- ・計算すべき総数の削減

五次魔方陣は、それを回転・対称移動して得られるものを同一視にすると2億7530万5224通り存在する。

A11	A12	A13	A14	A15
A21	A22	A23	A24	A25
A31	A32	A33	A34	A35
A41	A42	A43	A44	A45
A51	A52	A53	A54	A55

が魔方陣である時、上の赤い所の左右の列を交換しても縦横の和は変化しない。よって、斜めの和を等しくするために同様な操作を右2列、上2列、下2列に対して行うことによって新たな魔方陣を作ることが出来る(下の図の1個目)。同様なことを考え、上の図が魔方陣であるとき、次の3個の図も魔方陣となる事が分かる。

A22	A21	A23	A25	A24
A12	A11	A13	A15	A14
A32	A31	A33	A35	A34
A52	A51	A53	A55	A54
A42	A41	A43	A45	A44

A55	A52	A53	A54	A51
A25	A22	A23	A24	A21
A35	A32	A33	A34	A31
A45	A42	A43	A44	A41
A15	A12	A13	A14	A11

A44	A45	A43	A41	A42
A54	A55	A53	A51	A52
A34	A35	A33	A31	A32
A14	A15	A13	A11	A12
A24	A25	A23	A21	A22

なぜなら、縦横斜めの数字の組み合わせは変化していないためである。

よって、一番上の魔方陣を求めると自動的に下の3つも求めることができ、これらを同一視すると、求めるべき魔方陣の数は6882万6306通りになる。

0 から 24 までの数で構成されたある魔方陣に対して、24 からその魔方陣の要素を引いた数を新たにその要素とする表を作成すると、それはまた魔方陣となる。すべての魔方陣に対して、この方法で新たな魔方陣を作成できることから、これら 2 つの方陣を「双子」と呼ぶ事にする。この性質を用いて計算時間を短縮する事が本年度の取り組みである。

・アルゴリズム

プログラムでは、まずこの魔方陣の 25 個のマスそれぞれに 0 から 24 の数を入れ、5 桁の 25 進数が昇順に縦に 5 個並んでいるものとみなし、さらに 25 個全部で 25 桁の 25 進数とみなす。また、各行の 5 桁の 5 進数もまた各位の数が昇順でかつ各位の数の和が 60 になるようにする。この条件を満たす方陣を 25 桁の 25 進数として昇順に考えたものを魔方陣の元とする。例えば、昇順で最初に出てくる並び替えは次のようなものである。

0	1	12	23	24
2	3	13	20	22
4	5	11	19	21
6	7	14	15	18
8	9	10	16	17

(Base1)

2 番目が

0	1	12	23	24
2	3	13	20	22
4	5	11	19	21
6	7	14	16	17
8	9	10	15	18

3 番目が、

0	1	12	23	24
2	3	13	20	22
4	5	11	19	21
6	8	14	15	17
7	9	10	16	18

となる。

次にこの魔方陣の元の方陣 (A とする) に対してその双子 (B とする) を考える。B は各行とも降順になっているのでこれを昇順に並びかえる。さらに行同士を入れかえて 5 つの行が 5 桁の 25 進数として昇順に並ぶようにする。こうして B を魔方陣の元と同じ条件にする。これを C として、25 桁の 25 進数として A と C を比較する。A>C の場合、A 以前にすでに C があって、そこで魔方陣をカウントしていることになるので、この A については考えず、この次の魔方陣の種について考える。A<C の場合は、まだ A も C も出てきていないことになるので、A をもとに魔方陣を作成・登録すると同時にその双子魔方陣も登録する。A=C の場合、A 以前に C があったか否かが判断できないので、双子の存在は考えず A をもとに魔方陣を作成・登録するのみとする。

次に魔方陣の種から実際に魔方陣を作ることを考える。一番上の行を固定したまま、列の数の和が 60 になるように 2 行目以降の数をその行内で並び替える。たとえば上の Base 1 について 1 列目の和が 60 になるように並び替えを行うと次のようになる。

0	1	12	23	24
13	2	3	20	22
19	4	5	11	21
18	6	7	14	15
10	8	9	16	17

この列の並び換えは順序をつけて昇順で順次行う。行の内で並び換えを行うので、行の数の組み合わせは変わらず、その結果、横の和は 60 のままである。順序をつけて昇順で行うので、取りこぼしや重複も防ぐことができる。以下、再帰的に 2 列目、3 列目と順にその和を 60 にする。4 列目の和が 60 になると自動的に 5 列目の和も 60 になる。

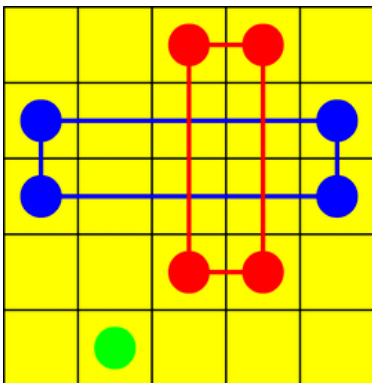
0	1	12	23	24
13	22	20	2	3
19	21	5	4	11
18	7	6	15	14
10	9	17	16	8

(Base2)

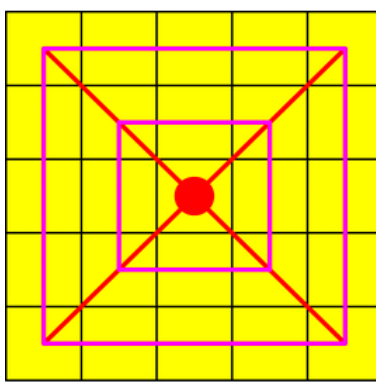
各列の和が 60 になると今度は縦方向について双子のチェックをする必要がある。まず、この方陣 (D とする) の行と列を入れかえる。次に各行内を昇順に並びかえる。各行を 5 桁の 25 進数とみなし、上から昇順になるように行どうしを入れかえる。この方陣を E とする。方陣 D の双子 (F とする) についても方陣 D と同じ操作を行い、これを G とする。E と G をともに 25 桁の 25 進数とみなす。E>G の場合、D を考える以前にすでに F が出てきていてカウント済みということになるので、この場合は無視して次の候補に進む。E<G の場合、E も G もまだ出てきていないことになるので、D をもとに次の操作に進む。E=G の場合は考慮する必要がないので無視する (後述)。

行と列それぞれの和がすべて 60 となったので、次は魔方陣の斜めの和が 60 になるように並び替える。ポイントはやはり各行と列の和がすでに 60 になっているという点である。例えば、上の Base2 の 1 行目と 2 行目を入れ替えてもその数の組み合わせは変化しないので各行・列の和は変化せずその和は 60 のままである。列同士を入れ替えても同様なので、行同士、列同士を自由に交換できることがわかる。その性質を利用して斜めの和が 60 になるように調整する。具体的には今までの操作で作った方陣の、各行から一つずつ列が重ならないように数を選択しその和が 60 となるような組み合わせを順序をつけてすべて検索する。5 次魔方陣であれば 5 つの要素をもつ斜めの数字の組は魔方陣に対し 2 つあり、その組のなかで一つは共通の数がある。また、その 2 つの斜めの組の要素は共通の数以外で組を作り、要素を頂点とする二つの正方形を作り出しているのとらえることができる。よって行がどのような順番で並んでいたとしても正方形の辺は平行移動のみをするのでそれらが長方形をつくることになる。

変形前



変形後



したがって、斜めの和が 60 になる候補から順序よく 2 つの候補を選び出し、上左図のように、共通な数が 1 つのみあり、残りの数 2 組ずつで長方形を形成するものを総当たりで検索する。該当する候補があれば、行どうし・列どうしの入れ替えを適切に行うことで上

右図のような 5 次魔方陣が完成する。例えば、上記過程を経て求められる魔方陣の一つが次のようなものである。

13	20	22	2	3
0	12	1	23	24
10	17	9	16	8
18	6	7	15	14
19	5	21	4	11

横方向の双子方陣のチェックで $A < C$ となり、かつ縦方向の双子方陣のチェックで $E < G$ となった場合、ここで求められた魔方陣に付随して双子の魔方陣が出てくることになるので、この魔方陣と双子の魔方陣両方を登録する。横方向の双子方陣のチェックで $A = C$ となった場合、 E と G の大小関係に関係なく双子方陣の存在は考えずここで求められた魔方陣のみを登録する。横方向の双子方陣のチェックで $A > C$ となった場合や、縦方向の双子方陣のチェックで $E > G$ となった場合はもちろん判定された時点で考えから外されているので問題ない。

したがって、このアルゴリズムは理論上重複や、漏れはあり得ない事が分かる。

・プログラムの実行方法

プログラムは方陣を表す 25 進数の上 8 桁の値毎に計算時間を求め、総計算時間を CPU のコアの数で割った時間より最小の差で長い時間ごとに計算領域を分割して作成し、176 個のコア一つでプログラム一つが実行されるようにした。そして、計算後計算結果を自動的に集計するプログラムを作成し、最長実行時間の CPU の計算時間を、5 次魔方陣の総数と全解それぞれを出力するときに対して計測した。使用したコンピュータの情報は以下の通りである。

場所	広島大学附属福山中・高等学校 情報処理演習室
コンピュータの台数	44 台
使用した OS	windows7 Pro
プロセッサ	Intel(R)Core(TM)i5-3570CPU@3.40GHz, 3401Mhz, 4 個のコア, 4 個のロジカルプロセッサ
実装メモリ	4.00GB
システムの種類	32 ビットオペレーティングシステム
HDD 容量	空き容量 85.9GB/108GB

・得られた結果

176 個のプログラムの計算時間は最大でも 28 秒であった。よって、目標である 30 秒以下で達成する事が出来た。

・考察

目標の計算時間 30 秒を切る事ができた。昨年度の 44 秒から大幅に時間短縮ができたことから双子魔方陣を利用することでかなりの効率化を実現することができたことがわかる。また、複数のコアに分散せず 1 台のコンピュータ・1 つのコアで計算を行うと、計算時間は 1 時間 19 分弱であったことから、パソコン 1 台で筑波大学のスーパーコンピュータより速い計算時間で計算出来た事もわかる。これはこのアルゴリズムで、計算する対象の絞り込みをより効率的に行った結果であると思われる。

176 個のプログラムの中には 22~23 秒で計算を終えるものも多くあったことから、計

算領域の分割をより効率的に行えば、あと1秒から2秒の計算時間の短縮は可能であったと考えられる。また、今回はデータの書き込みがHDDであったので、SDDなどデータの書き込み速度がより速いものを採用するとさらなる時間短縮が期待できるだろう。

話が変わって、6次魔方陣の総数の計算では、6次魔方陣の総数が約1800京と見積もられている。6次魔方陣には24個の同型が存在しているので、実質的には75京の魔方陣を見つければよいことになる。今回の5次魔方陣と同じペースで魔方陣が算出できたとして単純計算すると

$$75 \text{ 京} \div 6880 \text{ 万} \times 28 \text{ 秒} = \text{約 } 9700 \text{ 年}$$

となり計算できないことがわかる。よって、コンピュータのさらなる高速化が必要となるのはもちろんであるが、新たなる発想と枠組みによる更なる効率化が絶対不可欠である。

・結論

30秒以下で全5次魔方陣の全解を求める事が可能である。

・謝辞

甲斐章義先生にはプログラムの作成、情報処理研究室の使用などに関して多大な助言をいただきました。この場をもって感謝いたします。

・参考文献

<http://imrc.noip.me/article/kenkyukekka.pdf>

スーパーコンピューターを用いた五次魔方陣の全解の導出に関する pdf

「ランダムサンプリングによる6次魔方陣の総数の推定」数芸パズル第177号 1992年4月
pp13-20 大石

・付属データ

領域分割用計算領域計算時間対応表

0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24 (1)	0,1,16,20,23,2,10,11,13,24,3,4,12,19,22,5,6,14,17,18,7,8,9,15,21 (18)
0,1,12,23,24,2,7,16,17,18,3,4,10,21,22,5,6,14,15,20,8,9,11,13,19 (2)	0,1,16,21,22,2,5,14,15,24,3,4,10,20,23,6,7,11,17,19,8,9,12,13,18 (19)
0,1,13,22,24,2,6,9,20,23,3,4,14,18,21,5,7,12,17,19,8,10,11,15,16 (3)	0,1,16,21,22,2,9,12,13,24,3,4,10,20,23,5,6,14,17,18,7,8,11,15,19 (20)
0,1,13,22,24,2,9,12,14,23,3,4,15,17,21,5,6,10,19,20,7,8,11,16,18 (4)	0,1,17,18,24,2,5,9,21,23,3,4,11,20,22,6,7,12,16,19,8,10,13,14,15 (21)
0,1,14,21,24,2,5,8,22,23,3,4,15,18,20,6,7,11,17,19,9,10,12,13,16 (5)	0,1,17,18,24,2,8,13,14,23,3,4,10,21,22,5,7,9,19,20,6,11,12,15,16 (22)
0,1,14,21,24,2,8,11,16,23,3,4,12,19,22,5,7,10,18,20,6,9,13,15,17 (6)	0,1,17,19,23,2,4,9,21,24,3,5,10,20,22,6,7,13,16,18,8,11,12,14,15 (23)
0,1,14,22,23,2,3,17,18,20,4,5,6,21,24,7,8,10,16,19,9,11,12,13,15 (7)	0,1,17,19,23,2,7,13,14,24,3,4,10,21,22,5,6,11,18,20,8,9,12,15,16 (24)
0,1,14,22,23,2,7,9,18,24,3,4,12,20,21,5,6,13,17,19,8,10,11,15,16 (8)	0,1,17,19,23,2,13,14,15,16,3,4,7,22,24,5,6,8,20,21,9,10,11,12,18 (25)
0,1,14,22,23,2,10,15,16,17,3,4,8,21,24,5,6,11,18,20,7,9,12,13,19 (9)	0,1,17,20,22,2,7,8,19,24,3,4,9,21,23,5,6,15,16,18,10,11,12,13,14 (26)
0,1,15,20,24,2,6,9,21,22,3,4,11,19,23,5,7,13,17,18,8,10,12,14,16 (10)	0,1,17,20,22,2,11,14,15,18,3,4,6,23,24,5,7,8,19,21,9,10,12,13,16 (27)
0,1,15,20,24,2,9,12,14,23,3,4,10,21,22,5,6,13,17,19,7,8,11,16,18 (11)	0,1,18,19,22,2,7,10,17,24,3,4,9,21,23,5,6,13,16,20,8,11,12,14,15 (28)
0,1,15,21,23,2,4,17,18,19,3,5,6,22,24,7,8,9,16,20,10,11,12,13,14 (12)	0,1,18,19,22,2,13,14,15,16,3,4,6,23,24,5,7,10,17,21,8,9,11,12,20 (29)
0,1,15,21,23,2,8,10,16,24,3,4,11,20,22,5,6,12,18,19,7,9,13,14,17 (13)	0,1,18,20,21,2,7,13,14,24,3,4,8,22,23,5,9,10,17,19,6,11,12,15,16 (30)
0,1,16,19,24,2,3,17,18,20,4,5,6,22,23,7,8,9,15,21,10,11,12,13,14 (14)	0,2,12,22,24,1,3,17,18,21,4,5,8,20,23,6,7,13,15,19,9,10,11,14,16 (31)
0,1,16,19,24,2,7,10,18,23,3,4,11,20,22,5,6,13,15,21,8,9,12,14,17 (15)	0,2,12,22,24,1,9,15,16,19,3,4,10,20,23,5,6,11,17,21,7,8,13,14,18 (32)
0,1,16,19,24,2,11,14,15,18,3,4,8,22,23,5,6,12,17,20,7,9,10,13,21 (16)	0,2,13,21,24,1,7,9,20,23,3,4,12,19,22,5,6,14,17,18,8,10,11,15,16 (33)
0,1,16,20,23,2,6,13,15,24,3,4,10,21,22,5,7,11,18,19,8,9,12,14,17 (17)	0,2,13,21,24,1,10,15,16,18,3,4,8,22,23,5,6,12,17,20,7,9,11,14,19 (34)

0,2,13,22,23,1,6,12,17,24,3,4,14,18,21,5,7,9,19,20,8,10,11,15,16
(35)
0,2,13,22,23,1,10,11,14,24,3,4,12,20,21,5,6,15,16,18,7,8,9,17,19
(36)
0,2,14,20,24,1,5,17,18,19,3,4,8,22,23,6,7,10,16,21,9,11,12,13,15
(37)
0,2,14,20,24,1,9,12,15,23,3,4,10,21,22,5,6,13,17,19,7,8,11,16,18
(38)
0,2,14,21,23,1,5,13,17,24,3,4,11,20,22,6,7,10,18,19,8,9,12,15,16
(39)
0,2,14,21,23,1,8,16,17,18,3,4,7,22,24,5,6,10,19,20,9,11,12,13,15
(40)
0,2,15,19,24,1,5,9,22,23,3,4,12,20,21,6,7,13,16,18,8,10,11,14,17
(41)
0,2,15,19,24,1,8,13,16,22,3,4,9,21,23,5,6,11,18,20,7,10,12,14,17
(42)
0,2,15,20,23,1,4,14,17,24,3,5,9,21,22,6,7,10,18,19,8,11,12,13,16
(43)
0,2,15,20,23,1,8,12,17,22,3,4,10,19,24,5,7,9,18,21,6,11,13,14,16
(44)
0,2,15,21,22,1,4,16,19,20,3,5,10,18,24,6,7,11,13,23,8,9,12,14,17
(45)
0,2,15,21,22,1,8,13,14,24,3,4,10,20,23,5,6,12,18,19,7,9,11,16,17
(46)
0,2,16,18,24,1,4,15,17,23,3,5,9,21,22,6,7,8,19,20,10,11,12,13,14
(47)
0,2,16,18,24,1,8,11,17,23,3,4,10,21,22,5,6,14,15,20,7,9,12,13,19
(48)
0,2,16,19,23,1,4,14,17,24,3,5,9,21,22,6,7,12,15,20,8,10,11,13,18
(49)
0,2,16,19,23,1,8,14,15,22,3,4,9,20,24,5,6,10,18,21,7,11,12,13,17
(50)
0,2,16,20,22,1,5,11,19,24,3,4,9,21,23,6,7,12,17,18,8,10,13,14,15
(51)
0,2,16,20,22,1,9,11,15,24,3,4,12,18,23,5,6,13,17,19,7,8,10,14,21
(52)
0,2,17,18,23,1,5,14,16,24,3,4,10,21,22,6,7,8,19,20,9,11,12,13,15
(53)
0,2,17,18,23,1,9,15,16,19,3,4,7,22,24,5,6,8,20,21,10,11,12,13,14
(54)
0,2,17,19,22,1,6,11,18,24,3,4,9,21,23,5,7,12,16,20,8,10,13,14,15
(55)
0,2,17,19,22,1,11,12,13,23,3,4,8,21,24,5,6,14,15,20,7,9,10,16,18
(56)
0,2,17,20,21,1,7,15,18,19,3,4,6,23,24,5,8,9,16,22,10,11,12,13,14
(57)
0,2,18,19,21,1,5,7,23,24,3,4,11,20,22,6,8,13,16,17,9,10,12,14,15
(58)
0,2,18,19,21,1,9,14,16,20,3,4,6,23,24,5,7,11,15,22,8,10,12,13,17
(59)
0,3,12,21,24,1,6,16,17,20,2,4,9,22,23,5,7,11,18,19,8,10,13,14,15
(60)
0,3,12,22,23,1,6,9,20,24,2,4,14,19,21,5,7,13,17,18,8,10,11,15,16
(61)
0,3,12,22,23,1,10,14,15,20,2,4,9,21,24,5,6,13,17,19,7,8,11,16,18
(62)
0,3,13,20,24,1,6,15,16,22,2,4,10,21,23,5,7,11,18,19,8,9,12,14,17
(63)
0,3,13,20,24,1,10,15,16,18,2,4,9,22,23,5,6,11,17,21,7,8,12,14,19
(64)
0,3,13,21,23,1,7,9,19,24,2,4,12,20,22,5,6,14,17,18,8,10,11,15,16
(65)
0,3,14,19,24,1,2,13,21,23,4,5,9,20,22,6,7,12,17,18,8,10,11,15,16
(66)
0,3,14,19,24,1,7,13,16,23,2,4,11,21,22,5,6,12,17,20,8,9,10,15,18
(67)
0,3,14,20,23,1,4,9,22,24,2,5,13,19,21,6,7,12,17,18,8,10,11,15,16
(68)
0,3,14,20,23,1,8,12,15,24,2,4,11,21,22,5,6,13,17,19,7,9,10,16,18
(69)
0,3,14,21,22,1,5,13,17,24,2,4,11,20,23,6,7,10,18,19,8,9,12,15,16
(70)
0,3,14,21,22,1,9,15,16,19,2,4,7,23,24,5,6,11,18,20,8,10,12,13,17
(71)
0,3,15,18,24,1,6,10,20,23,2,4,11,21,22,5,7,12,17,19,8,9,13,14,16
(72)
0,3,15,18,24,1,10,11,16,22,2,4,12,19,23,5,6,8,20,21,7,9,13,14,17
(73)
0,3,15,19,23,1,6,16,17,20,2,4,8,22,24,5,7,9,18,21,10,11,12,13,14
(74)
0,3,15,20,22,1,2,10,23,24,4,5,11,19,21,6,7,12,17,18,8,9,13,14,16
(75)
0,3,15,20,22,1,8,12,16,23,2,4,9,21,24,5,6,13,17,19,7,10,11,14,18
(76)
0,3,16,17,24,1,5,11,20,23,2,4,13,19,22,6,7,8,18,21,9,10,12,14,15
(77)

0,3,16,17,24,1,9,10,18,22,2,4,11,20,23,5,6,13,15,21,7,8,12,14,19
(78)
0,3,16,18,23,1,6,12,17,24,2,4,11,21,22,5,7,9,19,20,8,10,13,14,15
(79)
0,3,16,19,22,1,2,10,23,24,4,5,12,18,21,6,7,13,14,20,8,9,11,15,17
(80)
0,3,16,19,22,1,8,12,15,24,2,4,10,21,23,5,6,11,18,20,7,9,13,14,17
(81)
0,3,16,20,21,1,5,17,18,19,2,4,7,23,24,6,8,9,15,22,10,11,12,13,14
(82)
0,3,16,20,21,1,11,14,15,19,2,4,7,23,24,5,6,9,18,22,8,10,12,13,17
(83)
0,3,17,18,22,1,8,14,16,21,2,4,7,23,24,5,6,10,19,20,9,11,12,13,15
(84)
0,3,17,19,21,1,6,14,15,24,2,4,9,22,23,5,7,10,18,20,8,11,12,13,16
(85)
0,3,18,19,20,1,4,9,22,24,2,5,13,17,23,6,7,10,16,21,8,11,12,14,15
(86)
0,3,18,19,20,1,10,13,14,22,2,4,7,23,24,5,6,11,17,21,8,9,12,15,16
(87)
0,4,11,22,23,1,7,14,17,21,2,3,12,19,24,5,6,13,16,20,8,9,10,15,18
(88)
0,4,12,20,24,1,3,17,18,21,2,5,8,22,23,6,7,13,15,19,9,10,11,14,16
(89)
0,4,12,21,23,1,5,14,16,24,2,3,13,20,22,6,7,10,18,19,8,9,11,15,17
(90)
0,4,12,21,23,1,10,14,15,20,2,3,9,22,24,5,6,13,17,19,7,8,11,16,18
(91)
0,4,13,19,24,1,7,11,18,23,2,3,12,21,22,5,6,14,15,20,8,9,10,16,17
(92)
0,4,13,20,23,1,3,11,21,24,2,5,12,19,22,6,7,14,15,18,8,9,10,16,17
(93)
0,4,13,20,23,1,9,15,16,19,2,3,10,21,24,5,6,14,17,18,7,8,11,12,22
(94)
0,4,13,21,22,1,7,14,15,23,2,3,11,20,24,5,6,12,18,19,8,9,10,16,17
(95)
0,4,14,18,24,1,5,9,22,23,2,3,15,19,21,6,7,10,17,20,8,11,12,13,16
(96)
0,4,14,18,24,1,9,12,15,23,2,3,13,20,22,5,6,11,17,21,7,8,10,16,19
(97)
0,4,14,19,23,1,7,10,18,24,2,3,12,21,22,5,6,13,16,20,8,9,11,15,17
(98)
0,4,14,20,22,1,5,12,18,24,2,3,11,21,23,6,7,13,15,19,8,9,10,16,17
(99)
0,4,14,20,22,1,11,12,13,23,2,3,10,21,24,5,6,15,16,18,7,8,9,17,19
(100)
0,4,15,17,24,1,7,12,18,22,2,3,11,21,23,5,6,10,19,20,8,9,13,14,16
(101)
0,4,15,18,23,1,5,8,22,24,2,3,14,20,21,6,7,11,17,19,9,10,12,13,16
(102)
0,4,15,18,23,1,11,13,14,21,2,3,9,22,24,5,6,10,19,20,7,8,12,16,17
(103)
0,4,15,19,22,1,9,11,16,23,2,3,10,21,24,5,6,12,17,20,7,8,13,14,18
(104)
0,4,15,20,21,1,7,13,16,23,2,3,9,22,24,5,6,12,18,19,8,10,11,14,17
(105)
0,4,16,17,23,1,6,12,19,22,2,3,10,21,24,5,7,13,15,20,8,9,11,14,18
(106)
0,4,16,18,22,1,5,12,19,23,2,3,10,21,24,6,7,13,14,20,8,9,11,15,17
(107)
0,4,16,19,21,1,3,9,23,24,2,5,11,20,22,6,7,12,17,18,8,10,13,14,15
(108)
0,4,16,19,21,1,11,13,15,20,2,3,8,23,24,5,6,9,18,22,7,10,12,14,17
(109)
0,4,17,18,21,1,10,13,14,22,2,3,8,23,24,5,7,9,19,20,6,11,12,15,16
(110)
0,4,17,19,20,1,9,14,15,21,2,3,8,23,24,5,6,11,16,22,7,10,12,13,18
(111)
0,5,10,22,23,1,8,9,18,24,2,3,14,20,21,4,6,15,16,19,7,11,12,13,17
(112)
0,5,11,21,23,1,6,15,16,22,2,3,12,19,24,4,7,14,17,18,8,9,10,13,20
(113)
0,5,12,19,24,1,3,16,17,23,2,4,11,21,22,6,7,9,18,20,8,10,13,14,15
(114)
0,5,12,20,23,1,6,15,16,22,2,3,10,21,24,4,7,13,17,19,8,9,11,14,18
(115)
0,5,12,21,22,1,4,16,19,20,2,3,8,23,24,6,7,14,15,18,9,10,11,13,17
(116)
0,5,13,18,24,1,3,14,19,23,2,4,11,21,22,6,7,10,17,20,8,9,12,15,16
(117)
0,5,13,18,24,1,9,14,15,21,2,3,10,22,23,4,6,11,19,20,7,8,12,16,17
(118)
0,5,13,19,23,1,8,15,16,20,2,3,9,22,24,4,6,11,18,21,7,10,12,14,17
(119)
0,5,13,20,22,1,8,12,15,24,2,3,11,21,23,4,6,14,17,19,7,9,10,16,18
(120)

0,5,14,17,24,1,6,16,18,19,2,3,10,22,23,4,7,8,20,21,9,11,12,13,15
(121)
0,5,14,18,23,1,4,9,22,24,2,3,15,19,21,6,7,10,17,20,8,11,12,13,16
(122)
0,5,14,19,22,1,4,10,21,24,2,3,12,20,23,6,7,13,16,18,8,9,11,15,17
(123)
0,5,14,20,21,1,4,8,23,24,2,3,15,18,22,6,7,11,17,19,9,10,12,13,16
(124)
0,5,15,16,24,1,3,14,19,23,2,4,11,21,22,6,7,9,18,20,8,10,12,13,17
(125)
0,5,15,16,24,1,9,13,14,23,2,3,12,21,22,4,6,11,19,20,7,8,10,17,18
(126)
0,5,15,17,23,1,9,13,16,21,2,3,11,20,24,4,6,10,18,22,7,8,12,14,19
(127)
0,5,15,18,22,1,11,12,13,23,2,3,10,21,24,4,6,14,16,20,7,8,9,17,19
(128)
0,5,15,19,21,1,11,12,13,23,2,3,9,22,24,4,6,14,16,20,7,8,10,17,18
(129)
0,5,16,18,21,1,4,8,23,24,2,3,13,20,22,6,7,11,17,19,9,10,12,14,15
(130)
0,5,16,19,20,1,6,12,17,24,2,3,10,22,23,4,7,13,15,21,8,9,11,14,18
(131)
0,5,17,18,20,1,7,12,16,24,2,3,10,22,23,4,6,14,15,21,8,9,11,13,19
(132)
0,6,9,22,23,1,8,12,15,24,2,3,14,20,21,4,5,16,17,18,7,10,11,13,19
(133)
0,6,10,21,23,1,8,14,15,22,2,3,11,20,24,4,5,16,17,18,7,9,12,13,19
(134)
0,6,11,20,23,1,9,13,15,22,2,3,10,21,24,4,5,14,18,19,7,8,12,16,17
(135)
0,6,11,21,22,1,12,13,14,20,2,3,8,23,24,4,5,15,17,19,7,9,10,16,18
(136)
0,6,12,19,23,1,5,15,17,22,2,3,10,21,24,4,7,11,18,20,8,9,13,14,16
(137)
0,6,12,20,22,1,8,13,14,24,2,3,11,21,23,4,5,15,17,19,7,9,10,16,18
(138)
0,6,13,17,24,1,8,9,19,23,2,3,12,21,22,4,5,15,16,20,7,10,11,14,18
(139)
0,6,13,18,23,1,9,10,16,24,2,3,12,21,22,4,5,14,17,20,7,8,11,15,19
(140)
0,6,13,20,21,1,2,15,18,24,3,4,8,22,23,5,7,12,17,19,9,10,11,14,16
(141)
0,6,14,16,24,1,4,17,18,20,2,3,10,22,23,5,7,8,19,21,9,11,12,13,15
(142)
0,6,14,17,23,1,4,10,21,24,2,3,13,20,22,5,7,11,18,19,8,9,12,15,16
(143)
0,6,14,18,22,1,9,10,16,24,2,3,11,21,23,4,5,12,19,20,7,8,13,15,17
(144)
0,6,15,16,23,1,5,8,22,24,2,3,14,20,21,4,7,12,18,19,9,10,11,13,17
(145)
0,6,15,18,21,1,3,9,23,24,2,4,12,20,22,5,7,13,16,19,8,10,11,14,17
(146)
0,6,15,19,20,1,7,14,16,22,2,3,8,23,24,4,5,12,18,21,9,10,11,13,17
(147)
0,6,16,18,20,1,5,8,22,24,2,3,11,21,23,4,7,13,17,19,9,10,12,14,15
(148)
0,6,17,18,19,1,10,14,15,20,2,3,8,23,24,4,5,13,16,22,7,9,11,12,21

(149)
0,7,9,21,23,1,5,15,17,22,2,3,11,20,24,4,6,13,18,19,8,10,12,14,16
(150)
0,7,10,20,23,1,9,13,15,22,2,3,12,19,24,4,5,14,16,21,6,8,11,17,18
(151)
0,7,11,19,23,1,6,13,16,24,2,3,12,21,22,4,5,14,17,20,8,9,10,15,18
(152)
0,7,12,17,24,1,3,11,22,23,2,4,13,20,21,5,6,14,16,19,8,9,10,15,18
(153)
0,7,12,18,23,1,11,15,16,17,2,3,9,22,24,4,5,10,20,21,6,8,13,14,19
(154)
0,7,12,20,21,1,10,14,16,19,2,3,8,23,24,4,5,11,18,22,6,9,13,15,17
(155)
0,7,13,16,24,1,11,14,15,19,2,3,10,22,23,4,5,12,18,21,6,8,9,17,20
(156)
0,7,13,18,22,1,12,14,16,17,2,3,8,23,24,4,5,10,20,21,6,9,11,15,19
(157)
0,7,14,15,24,1,6,13,17,23,2,3,12,21,22,4,8,9,19,20,5,10,11,16,18
(158)
0,7,14,17,22,1,5,10,20,24,2,3,11,21,23,4,6,13,18,19,8,9,12,15,16
(159)
0,7,14,19,20,1,6,11,18,24,2,3,10,22,23,4,5,13,17,21,8,9,12,15,16
(160)
0,7,15,18,20,1,3,16,17,23,2,4,8,22,24,5,6,9,19,21,10,11,12,13,14
(161)
0,7,16,18,19,1,9,10,17,23,2,3,11,20,24,4,5,8,21,22,6,12,13,14,15
(162)
0,8,10,19,23,1,2,16,17,24,3,4,11,20,22,5,6,13,15,21,7,9,12,14,18
(163)
0,8,11,18,23,1,6,16,17,20,2,3,9,22,24,4,7,13,15,21,5,10,12,14,19
(164)
0,8,12,16,24,1,3,17,18,21,2,4,9,22,23,5,6,10,19,20,7,11,13,14,15
(165)
0,8,12,19,21,1,2,16,17,24,3,4,10,20,23,5,6,9,18,22,7,11,13,14,15
(166)
0,8,13,15,24,1,9,12,16,22,2,3,11,21,23,4,5,14,17,20,6,7,10,18,19
(167)
0,8,13,18,21,1,9,14,16,20,2,3,10,22,23,4,5,12,15,24,6,7,11,17,19
(168)
0,8,14,18,20,1,3,10,22,24,2,4,12,19,23,5,6,11,17,21,7,9,13,15,16
(169)
0,8,16,17,19,1,5,9,21,24,2,3,10,22,23,4,6,12,18,20,7,11,13,14,15
(170)
0,9,11,17,23,1,3,10,22,24,2,4,13,20,21,5,6,12,18,19,7,8,14,15,16
(171)
0,9,12,15,24,1,8,10,18,23,2,3,13,20,22,4,5,11,19,21,6,7,14,16,17
(172)
0,9,13,14,24,1,5,11,20,23,2,3,12,21,22,4,6,15,16,19,7,8,10,17,18
(173)
0,9,13,18,20,1,5,14,16,24,2,3,10,22,23,4,6,12,17,21,7,8,11,15,19
(174)
0,9,16,17,18,1,5,15,19,20,2,3,8,23,24,4,6,7,21,22,10,11,12,13,14
(175)
0,10,12,16,22,1,4,8,23,24,2,3,14,20,21,5,6,13,17,19,7,9,11,15,18
(176)
0,12,13,14,21,1,2,10,23,24,3,4,11,20,22,5,6,15,16,18,7,8,9,17,19
counter=68826306 finish 2015/7/31 21:9:27start 2015/7/31 19:48:45

魔方陣全解算出プログラム

c:/Python27/python.exe
-*- coding: utf-8 -*-

import datetime

```
def sort(c=[i for i in range(25)],k=1):
    for i in range(k,25):
        for j in range(i+1,25):
            if c[i]>c[j]:
                temp=c[i]
                c[i]=c[j]
                c[j]=temp
    return c
```

```
def sort5(x=[i for i in range(5)],k=1):
    for i in range(k,5):
        for j in range(i+1,5):
            if x[i]>x[j]:
                temp=x[i]
                x[i]=x[j]
                x[j]=temp
    return x
```

```
def inc(c=[i for i in range(25)],k=23,l=1):
    if c[k]+l>=25:
```

```
c = inc(c=c,k=k-1,l=1)
```

```
c=sort(c=c,k=k)
```

```
return c
```

```
else:
```

```
re=0
```

```
for i in range(k+1,25):
```

```
    if c[k]+l == c[i]:
```

```
        d = c[k]
```

```
        c[k] = c[i]
```

```
        c[i] = d
```

```
        if k<=22:
```

```
            c=sort(c=c,k=k+1)
```

```
            re=1
```

```
            break
```

```
if re==0:
```

```
    return inc(c=c,k=k,l=l+1)
```

```
else:
```

```
    return c
```

```
def inc5(x=[i for i in range(5)],k=3,l=1):
```

```
    if k<0:
```

```
        return [0 for i in range(5)]
```

```
    elif x[k]+l>=5:
```

```
        x=inc5(x=x,k=k-1,l=1)
```

```
        x=sort5(x=x,k=k)
```



```

return x
else:
    re=0
    for i in range(k+1,5):
        if x[k]+1== x[i]:
            d = x[k]
            x[k] = x[i]
            x[i] = d
            if k<=2:
                x=sort5(x=x,k=k+1)
            re=1
            break
    if re==0:
        return inc5(x=x,k=k,l=l+1)
    else:
        return x

def replace(c=[i for i in range(25)]):
    i=0
    while i<5:
        re=-1
        if i==0:
            while c[0]+c[1]+c[2]+c[3]+c[4]<60 or c[0]>c[1] or c[1]>c[2] or c[2]>c[3]
or c[3]>c[4]:
                if c[0]>c[1] or c[0]+c[1]+c[2]*3>57:
                    c=inc(c=c,k=1)
                elif c[1]>c[2] or c[0]+c[1]+c[2]+c[3]*2>59:
                    c=inc(c=c,k=2)
                elif c[2]>c[3]:
                    c=inc(c=c,k=3)
                else:
                    c=inc(c=c,k=4)
            elif i==1:
                while c[5]+c[6]+c[7]+c[8]+c[9]<60 or c[5]>c[6] or c[6]>c[7] or c[7]>c[8]
or c[8]>c[9] or c[0]>c[5]:
                    if c[0]>c[5] or c[5]+c[6]*4>54:
                        c=inc(c=c,k=4)
                    elif c[5]>c[6] or c[5]+c[6]+c[7]*3>57:
                        c=inc(c=c,k=6)
                    elif c[6]>c[7] or c[5]+c[6]+c[7]+c[8]*2>59:
                        c=inc(c=c,k=7)
                    elif c[7]>c[8]:
                        c=inc(c=c,k=8)
                    else:
                        c=inc(c=c,k=9)
            else:
                while c[5*i]+c[5*i+1]+c[5*i+2]+c[5*i+3]+c[5*i+4]<60 or
c[5*i]>c[5*i+1] or c[5*i+1]>c[5*i+2] or c[5*i+2]>c[5*i+3] or c[5*i+3]>c[5*i+4]
or c[5*(i-1)]>c[5*i]:
                    if c[5*(i-1)]>c[5*i] or c[5*i]+c[5*i+1]*4>54:
                        c=inc(c=c,k=5*i-1)
                    elif c[5*i]>c[5*i+1] or c[5*i]+c[5*i+1]+c[5*i+2]*3>57:
                        c=inc(c=c,k=5*i+1)
                    elif c[5*i+1]>c[5*i+2] or c[5*i]+c[5*i+1]+c[5*i+2]+c[5*i+3]*2>59:
                        c=inc(c=c,k=5*i+2)
                    elif c[5*i+2]>c[5*i+3]:
                        c=inc(c=c,k=5*i+3)
                    else:
                        c=inc(c=c,k=5*i+4)
            if i>=1:
                if c[0]>c[5] or c[0]+c[1]+c[2]+c[3]+c[4]<60:
                    re=0
                else:
                    for j in range(i):
                        if j==0:
                            if c[0]>c[1] or c[1]>c[2] or c[2]>c[3] or c[3]>c[4] or
c[0]+c[1]+c[2]+c[3]+c[4]<60:
                                re=0
                                break
                            else:
                                if c[5*j-5]>c[5*j] or c[5*j]>c[5*j+1] or c[5*j+1]>c[5*j+2] or
c[5*j+2]>c[5*j+3] or c[5*j+3]>c[5*j+4] or
c[5*j]+c[5*j+1]+c[5*j+2]+c[5*j+3]+c[5*j+4]<60:
                                    re=j
                                    break
                        if re==1:
                            i+=1
                        else:
                            i=re
                    return c

def maketwin1(c=[i for i in range(25)]):
    d=[0 for i in range(25)]
    x=[0 for i in range(5)]
    for i in range(5):
        for j in range(5):
            d[5*i+j]=24-c[5*i+4-j]
            for i in range(5):
                for j in range(i+1,5):
                    if d[5*i]>d[5*j]:
                        for k in range(5):
                            x[k]=d[5*i+k]
                            d[5*i+k]=d[5*j+k]
                            d[5*j+k]=x[k]
            return d

def maketwin2(c=[i for i in range(5)] for i in range(5)):
    d=[[0 for j in range(5)] for i in range(5)]
    for i in range(5):
        for j in range(5):
            d[i][j]=24-c[i][j]
    return d

def twincompare(c=[i for i in range(25)],d=[i for i in range(25)]):
    i=0
    k=1
    for i in range(25):
        if c[i]>d[i]:
            k=0
            break
        elif c[i]<d[i]:
            k=2
            break
    return k

def validatetwin(c=[i for i in range(25)],e=[i for i in range(5)] for i in
range(5)):
    r=[i for i in range(25)]
    ev=[[5*i+j for j in range(5)] for i in range(5)]
    for i in range(5):
        for j in range(5):
            ev[j][i]=e[i][j]
    for i in range(5):
        ev[i]=sort5(ev[i],k=0)
    for i in range(5):
        for j in range(i+1,5):
            if ev[i][0]>ev[j][0]:
                temp=ev[i]
                ev[i]=ev[j]
                ev[j]=temp
    for i in range(5):
        for j in range(5):
            r[5*i+j]=ev[i][j]
    tm=twincompare(c=c,d=r)
    return tm

def replace2(c=[i for i in range(25)],k=0,times=1):
    a=[[0 for j in range(5)] for i in range(5)]
    for i in range(5):
        for j in range(5):
            a[i][j]=c[5*i+j]
    k=colreplace(c1=c,c=a,k=k,times=times)
    return k

def colreplace(c1=[i for i in range(25)],c=[[5*i+j for j in range(5)] for i in
range(5)],s=0,k=0,times=1):
    a=[[5*i+j for j in range(5)] for i in range(5)]
    for i in range(5):
        for j in range(5):
            a[i][j]=c1[i][j]
    r=5-s
    i=1
    t=[0 for j in range(5)]
    while i<=4:
        b=i
        for j in range(4):
            t[4-j]=b*r
            b=int(b/r)
        if a[0][s]+a[1][s+t[1]]+a[2][s+t[2]]+a[3][s+t[3]]+a[4][s+t[4]]==60:
            for j in range(1,5):
                temp=a[j][s+t[j]]
                a[j][s+t[j]]=a[j][s]
                a[j][s]=temp
            a[j]=sort5(x=a[j],k=s+1)
            if s<=3:
                #print "colreplace !! -----"
                #pprt(p=a)
                if a[0][1]<a[1][0] and a[0][1]<a[2][0] and a[0][1]<a[3][0] and
a[0][1]<a[4][0]:
                    e=maketwin2(a)

```