Appendix to the note "On the Inoue invariants of the puzzles of Sudoku type"

Tetsuo Nakano, Kenji Arai, Hiromasa Watanabe

2013/3/27

In this appendix, we summarize several programs, their output logs and some detailed data on our main results.

1 Programs for 4-doku (details on Theorem 24)

The following program computes the number of the 4-doku puzzles with 4 clues which has No.1 board as the unique solution, and among them the number of the puzzles with the trivial Inoue invariant.

Program 1 (IS1_4.txt)

```
////////////////////////////////////////////////////////////////////
//This program consists of 2 programs Inoue4 and IS1_4.
//Inoue4 is the Inoue solver (algorithm) for 4-doku.
//Inou4([generators of the ideal]) outputs the pair (the set of solutions, Inoue invariant).
//If you type "IS1_4();", then IS1_4 computes both (i) the number of
//4-doku puzzles wit the 4 clues which has No.1 board as a solution with
//a unique solution and (ii) the number of puzzles among (i) with the
//trivial Inoue invariant.
////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////
////// basic settings for the coefficient ring (F_2)^4 ////////////

M2<e1,e2,e3,e4> := PolynomialRing(GF(2),4);
I2:= ideal<M2| e1^2-e1,e2^2-e2,e3^2-e3,e4^2-e4,e1*e2,e1*e3,e1*e4,
e2*e3,e2*e4,e3*e4>;
M<e1,e2,e3,e4> := M2/I2;

P<a44,a43,a42,a41,a34,a33,a32,a31,a24,a23,a22,a21,a14,a13,a12,a11>:=PolynomialRing(M,16);


////////////////////////////////////////////////////////////////////
////////////////// program ASPTransform (= SBGTrans) //////////////////
```

```
forward AlmostSolution3, SBG2, Const;  /// 3 subroutines are called

SBGTrans := function(F)
if Const(F) ne 0 then return F;
end if;

G:= F;
check := 0;
n:= [1,1,1,1];

while n[2]+ n[3]+n[4] ne 0 and check eq 0 do

n,G := AlmostSolution3(G);

G:= SBG2(G);

if Const(G) ne 0 then check := 1;
end if;

end while;

return G;

end function;

////////////////////////////////////////////////////
///////// AlmostSolution3 ////////////////////////



AlmostSolution3 := function(F);

X := [e1,e2,e3,e4];
Y := [a11,a12,a13,a14,a21,a22,a23,a24,
      a31,a32,a33,a34,a41,a42,a43,a44];
E := e1+e2+e3+e4;
Y1:= [E*y: y in Y];

G := F;
Zero := [i*0:i in [1..16]];
Z1 := [E*m2+m1 : m1 in X,m2 in Y];
Z2 := [(E+m1)*m2:m1 in X,m2 in Y];

S := [m : m in G | m in Z1]; // the set of solution polynomials
S1 := [m : m in G | m in Z2];// the set of AS polynomial of type 1
S3 := S cat S1;

W := SequenceToSet(G) diff SequenceToSet(S3);
G := SetToSequence(W); ///the set of SBG bases with solution polynomials and AS
                        ///polynomials of type 1 removed


G0:=[ g: g in G | LeadingTerm(g) in Y1];
S0:= [];
T0:= [];

for g in G0 do
```

```
U0:= [e: e in X | e*(g-LeadingTerm(g)) eq g - LeadingTerm(g)];
if #U0 eq 1 then S0:= S0 cat [g]; T0 := T0 cat [LeadingTerm(g)+U0[1]];
end if;
end for;



T2 := [];
S2:= [];

for i in [1..#G] do
check := 0;
        for j in [1..4] do
if X[j]*G[i] eq X[j]*LeadingTerm(G[i])+X[j] and LeadingMonomial(G[i]) in Y
        then
                        check := check +1;
T2 := T2 cat [E*LeadingMonomial(G[i])+X[j]];
end if;
end for;
if check ne 0 then S2:= S2 cat [G[i]]; end if;
end for;


T1 := [E*LeadingMonomial(m) + (E + LeadingCoefficient(m)) : m in S1];
//// transform AS polynomials of type 1 to solution polynomials
n := [#S,#S2,#S1,#S0];
S3 :=S2 cat S1;

G := F cat T1 cat T2 cat T0; /// add T0, T1 and T2 to F

return n, G;

end function;

////////////////////////////////////////////////////////////////////////////
////////////// SBG2 (computes the SBG basis of the ideal) //////////////////
////////////////////////////////////////////////////////////////////////////

////////// main body of SBG2 ////////////////////

forward Seibun, Boolsudoku, SBGroebner; /// 3 subroutines are called

SBG2:= function(F)

S := Seibun(F);
G := Boolsudoku(S);
G := SBGroebner(G);

return(Sort(G));

end function;

//////////////////////////////////////////////////

Seibun:= function(F)

C := [[],[],[],[]];
```

```
X := [e1,e2,e3,e4];

for t in [1..4] do
S1 := [m*(P!X[t]) : m in F];

A := [Coefficients(f) : f in S1| f ne 0];
B := [Monomials(f) : f in S1 | f ne 0];
A1:=[];
    for i in [1..#A] do
A1 := A1 cat [[Evaluate(m,X[t],1):m in A[i]]];
    end for;

for i in [1..#A1] do
C[t] := C[t] cat [&+[P!(A1[i][j]*B[i][j]):j in [1..#A1[i]]]];
end for;
end for;

P2<a44,a43,a42,a41,a34,a33,a32,a31,a24,a23,a22,a21,a14,a13,a12,a11>:=BooleanPolynomialRing(16);

C1:= [P2 ! m : m in C[1]];
C2:= [P2 ! m : m in C[2]];
C3:= [P2 ! m : m in C[3]];
C4:= [P2 ! m : m in C[4]];

C := [C1,C2,C3,C4];

return C;

end function;


//////////////////////////////////////////////////////

Boolsudoku := function(F);

G1 := GroebnerBasis(F[1]);
G2 := GroebnerBasis(F[2]);
G3 := GroebnerBasis(F[3]);
G4 := GroebnerBasis(F[4]);

G11 := [e1*(P!m) : m in G1];
G12 := [e2*(P!m) : m in G2];
G13 := [e3*(P!m) : m in G3];
G14 := [e4*(P!m) : m in G4];

G20 := G11 cat G12 cat G13 cat G14;

return G20;
end function;

////////////////////////////////////////////

SBGroebner := function(G)

X := {@ LeadingMonomial(m) : m in G @};
n := #X;
Y := [i*(P ! 0) : i in [1..n]];
```

```
for i in [1..n] do
    Y[i] := &+[x : x in G | LeadingMonomial(x) eq X[i]];
end for;

return Y;
end function;

///////////////// end of SBG2 ////////////////////
////////////////////////////////////////////////


Const := function(F);
Zero := [i*0:i in [1..16]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;



//////////////////////////////////////////////////
////////////// Inoue algorithm ///////////////////////
//////////////////////////////////////////////////

//////////////////////////////////////////////////
////////// Inoue1 (main body of Inoue algorithm)/////

forward Solpoly, variable, Const, inoue1;

inoue1 := function(F: d := 0)

L := [];
D := [];
I := [0,0,0];
X := [e1,e2,e3,e4];
E := e1+e2+e3+e4;
Zero := [i*0:i in [1..16]];

G:= SBGTrans(F);

S:= Solpoly(G);


d:= d+1;
D := D cat [d];

if #S eq #G then L := L cat [G]; I[1]:= I[1]+1; // a solution leaf is added
end if;

if Const(G) ne 0 then L := L cat [];I[1]:= I[1]+1;// a non solution leaf is added
end if;

if #S lt #G and Const(G) eq 0 then
V := variable(G);
G2:= G;

for s in S do
```

```
G2:= [Evaluate(f,LeadingMonomial(s),TrailingTerm(s)) : f in G2];
G2 := [m : m in G2 | m ne 0];
end for;

G3 := [g : g in G2 | IsUnivariate(g,V[1])];

if G3 eq [] then X:= X;
end if;

if G3 ne [] then
a:= #G3;
zero:= [P!(i*0) : i in [1..a]];
A := [x : x in X | [Evaluate(f,V[1],x): f in G3] eq zero];
if A eq [] then L := L cat []; I[1]:= I[1]+1; // one more leaf

else X:= A;
end if;
end if;

K :=[G cat [E*V[1]+x]: x in X];
K1:= [SBG2(K[i]): i in [1..#K]];
K2:= [m : m in K1 | Const(m) eq 0];
if #K2 eq 0 then L := L cat [];  I[1]:= I[1]+1;

else I[3]:= I[3]+1; // one more node
I[2]:= I[2] + #K2; // branch number increased by #K2
end if;


for m in K2 do
L1,I1,D1:= inoue1(m: d:= d);
L := L cat L1;
D := D cat D1;
I[1]:= I[1]+ I1[1];I[2]:= I[2]+I1[2];I[3]:= I[3]+I1[3];
end for;
end if;

return L,I,D;

end function;

//////// end of Inoue1 ////////////////////////
////////////////////////////////////////////////////////

Solpoly := function(F); ///output the set of solution polynomials
                        ///contained in the SBG bases F

X := [e1,e2,e3,e4];
E := e1+e2+e3+e4;
Y := [E*a11,E*a12,E*a13,E*a14,E*a21,E*a22,E*a23,E*a24,
      E*a31,E*a32,E*a33,E*a34,E*a41,E*a42,E*a43,E*a44];

E := e1+e2+e3+e4;
S := [m : m in F | LeadingTerm(m) in Y and #Terms(m) eq 2 and TrailingTerm(m) in X];
     // the set of solution polynomials

return S;
```

```
end function;

////////////////////////////////

variable := function(F);
Y := [a11,a12,a13,a14,a21,a22,a23,a24,
      a31,a32,a33,a34,a41,a42,a43,a44];

S := Solpoly(F);
K := [LeadingMonomial(m) : m in S];
K1 := [m : m in Y | m notin K];
return K1;

end function;


/////////////////////////////////////////
Const := function(F);
Zero := [i*0:i in [1..16]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;

/////////////////////////////////////////////
/////////////////////////////////////////////


matrix4 := function(F);

Z := IntegerRing();
X1 := [e1,e2,e3,e4];
X2 := [Z!1,Z!2,Z!3,Z!4];
W1:=[i*0 : i in [1.16]];

for i in [1..16] do

F[i]:=TrailingTerm(F[i]);

if F[i] eq e1 then W1[i]:=X2[1];
end if;

if F[i] eq e2 then W1[i]:=X2[2];
end if;

if F[i] eq e3 then W1[i]:=X2[3];
end if;

if F[i] eq e4 then W1[i]:=X2[4];
end if;

end for;

W2 := [Z!m : m in W1];
W3 := [Matrix(4,4,W2)];

return W3;
```

```
end function;


////////////////////////////////
////////////////////////////////

inoue11 := function(F)

F:= SBG2(F);
L,I,D:= inoue1(F);

return L,I,D;
end function;

////////////////////////////////////////////////////////////
///////Inoue4 (Inoue solver for 4-doku) ////////////////////
/// input= ideal of the puzzle /////////////////////////////
//// output = (the set of solutions, Inoue invariant) //////
////////////////////////////////////////////////////////////

Inoue4 := function(L);
Q3:=[];
I2:=[0,0,0];
Q1,I,D:=inoue11(L);
for i in [1..#Q1] do
Q2:=matrix4(Q1[i]);
Q3:= Q3 cat Q2;
end for;
I[2]:= I[2]+1; I[3]:= I[3]+1;
DM:=Max(D);
I2[1]:=I[1]+I[3]; I2[2]:=I[1]; I2[3]:=DM;

return Q3,I2;

end function;



////////////////////////////////////////////////////////////
//////////////IS1_4 //////////////////
/////////This program computes both (i) the number of 4-doku
//////// puzzles with 4 clues which has No.1 board as the unique solution
/////// and (ii) the number of puzzles with the trivial Inoue invariant
/////// among (i).
////// Usage: Type IS1_4(); after the Magma prompt and this program runs

IS1_4:= procedure();

X:=[a11+e1,a12+e2,a13+e3,a14+e4,a21+e3,a22+e4,a23+e1,a24+e2,a31+e2,a32+e1,
a33+e4,a34+e3,a41+e4,a42+e3,a43+e2,a44+e1]; // No.1 solution board
////// modify this line in the other cases

E:= e1+e2+e3+e4;


F:=[
a11+a12+a13+a14+E,a11*a12,a11*a13,a11*a14,a12*a13,a12*a14,a13*a14,
```

```
a21+a22+a23+a24+E,a21*a22,a21*a23,a21*a24,a22*a23,a22*a24,a23*a24,
a31+a32+a33+a34+E,a31*a32,a31*a33,a31*a34,a32*a33,a32*a34,a33*a34,
a41+a42+a43+a44+E,a41*a42,a41*a43,a41*a44,a42*a43,a42*a44,a43*a44,

a11+a21+a31+a41+E,a11*a21,a11*a31,a11*a41,a21*a31,a21*a41,a31*a41,
a12+a22+a32+a42+E,a12*a22,a12*a32,a12*a42,a22*a32,a22*a42,a32*a42,
a13+a23+a33+a43+E,a13*a23,a13*a33,a13*a43,a23*a33,a23*a43,a33*a43,
a14+a24+a34+a44+E,a14*a24,a14*a34,a14*a44,a24*a34,a24*a44,a34*a44,

a11^2+a11,a12^2+a12,a13^2+a13,a14^2+a14,
a21^2+a21,a22^2+a22,a23^2+a23,a24^2+a24,
a31^2+a31,a32^2+a32,a33^2+a33,a34^2+a34,
a41^2+a41,a42^2+a42,a43^2+a43,a44^2+a44,

a11+a12+a21+a22+E,a11*a22,a12*a21,
a31+a32+a41+a42+E,a31*a42,a32*a41,
a13+a14+a23+a24+E,a13*a24,a14*a23,
a33+a34+a43+a44+E,a33*a44,a34*a43];


count1 := 0;
count2 := 0;


///// modify the for part below in the other cases

for n4 in [1..13] do
for n3 in [n4+1..14] do
for n2 in [n3+1..15] do
for n1 in [n2+1..16] do


L:=[X[n4],X[n3],X[n2],X[n1]];

L:= F cat L;

W,I2 := Inoue4(L);

if #W eq 1 then
count1 :=count1 +1;
count1;

if #W eq 1 and I2 eq [2,1,1] then
count2 := count2 +1;
count2;

end if;
end if;

end for;
end for;
end for;
end for;

"the number of puzzles with a unique solution =", count1;
"the number of puzzles with a unique solution and the trivial Inoue invariant = ", count2;
```

```
end procedure;

///////////////// end of IS1_4/////////////////////////////////////////
////////////////////////////////////////////////////
```

The following is the output of the program IS1_4.

OutputLog 2

```
Magma V2.18-3     Fri Mar 08 2013 10:27:12 on vostro   [Seed = 1062686776]
Type ? for help.  Type <Ctrl>-D to quit.
> load "IS1_4.txt";
Loading "C:\Program Files (x86)\Magma\libs\examples\IS1_4.txt"
> time IS1_4();
1
1
2
2
3
3
4
4
5
5
6
6
7
7
8
8
9
9
10
10
11
11
12
12
the number of puzzles with a unique solution = 12
the number of puzzles with a unique solution and the trivial Inoue invariant = 12

Time: 123.241 (seconds)
>
```

From this output, we find that there are 12 puzzles with the unique solution No.1 with 4 clues and all of them have the trivial Inoue invariant. For the complete proof of Theorem 24, just do the same computation for both No.1 and No.2 boards with k clues ($k = 4, 5, 6$). You need to modify the last part (procedure IS1_4) of Program 1 suitably.

We also compute an example of 4-doku puzzles by Inoue4.

Example 3

Let's compute the solutions and the Inoue invariant of the following 4-doku puzzle.

Table 1: An example of 4-doku puzzle

1			
		3	
	2		

```
> load "IS1_4.txt";
Loading "C:\Program Files (x86)\Magma\libs\examples\IS1_4.txt"
> load "Input_4.txt";
Loading "C:\Program Files (x86)\Magma\libs\examples\Input_4.txt"
> time Inoue4(L1);
[
    [1 3 2 4]
    [2 4 3 1]
    [4 2 1 3]
    [3 1 4 2],

    [1 3 4 2]
    [2 4 3 1]
    [3 2 1 4]
    [4 1 2 3],

    [1 3 4 2]
    [2 4 3 1]
    [4 2 1 3]
    [3 1 2 4]
]
[ 6, 3, 3 ]
Time: 0.062 (seconds)
>
```

This computations shows this puzzle has three solutions as above and its Inoue invariant is $(6, 3, 3)$.

The Input_4.txt file is as follows.

Program 4 (Input_4.txt)

```
b:= e1+e2+e3+e4;

F := [a11+a12+a13+a14+b,a11*a12,a11*a13,a11*a14,a12*a13,a12*a14,a13*a14,
a21+a22+a23+a24+b,a21*a22,a21*a23,a21*a24,a22*a23,a22*a24,a23*a24,
a31+a32+a33+a34+b,a31*a32,a31*a33,a31*a34,a32*a33,a32*a34,a33*a34,
a41+a42+a43+a44+b,a41*a42,a41*a43,a41*a44,a42*a43,a42*a44,a43*a44,
```

```
a11+a21+a31+a41+b,a11*a21,a11*a31,a11*a41,a21*a31,a21*a41,a31*a41,
a12+a22+a32+a42+b,a12*a22,a12*a32,a12*a42,a22*a32,a22*a42,a32*a42,
a13+a23+a33+a43+b,a13*a23,a13*a33,a13*a43,a23*a33,a23*a43,a33*a43,
a14+a24+a34+a44+b,a14*a24,a14*a34,a14*a44,a24*a34,a24*a44,a34*a44,
a11+a12+a21+a22+b,a11*a12,a11*a21,a11*a22,a12*a21,a12*a22,a21*a22,
a13+a14+a23+a24+b,a13*a14,a13*a23,a13*a24,a14*a23,a14*a24,a23*a24,
a31+a32+a41+a42+b,a31*a32,a31*a41,a31*a42,a32*a41,a32*a42,a41*a42,
a33+a34+a43+a44+b,a33*a34,a33*a43,a33*a44,a34*a43,a34*a44,a43*a44,
a11^2+a11,a12^2+a12,a13^2+a13,a14^2+a14,
a21^2+a21,a22^2+a22,a23^2+a23,a24^2+a24,
a31^2+a31,a32^2+a32,a33^2+a33,a34^2+a34,
a41^2+a41,a42^2+a42,a43^2+a43,a44^2+a44];

L1:= [a11+e1,a23+e3,a32+e2];
L1:= F cat L1;
```

---

2 Programs for diagonal 5-doku

2.1 Details on the proof of Proposition 27

The following program is the Inoue solver for diagonal 5-doku puzzles.

Program 5 (Inoue5.txt)

```
//////////////// Inoue Solver for daiagonal 5-doku//////////////////////////
////////////////////////////////////////////////////////////////////////

M2<e1,e2,e3,e4,e5> := PolynomialRing(GF(2),5);
I2:= ideal<M2| e1^2-e1,e2^2-e2,e3^2-e3,e4^2-e4,e5^2-e5,e1*e2,e1*e3,e1*e4,e1*e5,
e2*e3,e2*e4,e2*e5,e3*e4,e3*e5,
e4*e5>;
M<e1,e2,e3,e4,e5> := M2/I2;

P<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,a15,a14,a13,a12,a11>:=PolynomialRing(M,25);


////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////

forward AlmostSolution3, SBG2, Const;

SBGTrans := function(F)

if Const(F) ne 0 then return F;
end if;

G:= F;
check := 0;
n:= [1,1,1,1];

while n[2]+ n[3]+n[4] ne 0 and check eq 0 do
```

```
n,G := AlmostSolution3(G);

G:= SBG2(G);

if Const(G) ne 0 then check := 1;
end if;

end while;

return G;

end function;

////////////////////////////////////////////////////



AlmostSolution3 := function(F);

X := [e1,e2,e3,e4,e5];
Y := [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];
E := e1+e2+e3+e4+e5;
Y1:= [E*y: y in Y];

G := F;
Zero := [i*0:i in [1..25]];
Z1 := [E*m2+m1 : m1 in X,m2 in Y];
Z2 := [(E+m1)*m2:m1 in X,m2 in Y];

S := [m : m in G | m in Z1]; // the set of solution polynomials
S1 := [m : m in G | m in Z2];// the set of AS polynomial of type 1
S3 := S cat S1;
// #F;

W := SequenceToSet(G) diff SequenceToSet(S3);
G := SetToSequence(W); ///the set of SBG bases with solution polynomials and
                       ///ASP of type 1 removed


G0:=[ g: g in G | LeadingTerm(g) in Y1];
S0:= [];
T0:= [];

for g in G0 do
U0:= [e: e in X | e*(g-LeadingTerm(g)) eq g - LeadingTerm(g)];
if #U0 eq 1 then S0:= S0 cat [g]; T0 := T0 cat [LeadingTerm(g)+U0[1]];
end if;
end for;



T2 := [];
S2:= [];
```

```
for i in [1..#G] do
check := 0;
        for j in [1..5] do
if X[j]*G[i] eq X[j]*LeadingTerm(G[i])+X[j] and LeadingMonomial(G[i]) in Y then
                        check := check +1;
T2 := T2 cat [E*LeadingMonomial(G[i])+X[j]];
end if;
end for;
if check ne 0 then S2:= S2 cat [G[i]]; end if;
end for;

T1 := [E*LeadingMonomial(m) + (E + LeadingCoefficient(m)) : m in S1];
      // transform AS polynomials of type 1 to solution polynomials
n := [#S,#S2,#S1,#S0];
S3 :=S2 cat S1;

G := F cat T1 cat T2 cat T0; /// add T1 and T2 to F, not F-(S1 +S2)

return n, G;

end function;

////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////
forward Seibun, Boolsudoku, SBGroebner;

SBG2:= function(F)

S := Seibun(F);
G := Boolsudoku(S);
G := SBGroebner(G);

return(Sort(G));

end function;

////////////////////////////////////////////////////////////

Seibun:= function(F)

C := [[],[],[],[],[]];
X := [e1,e2,e3,e4,e5];

for t in [1..5] do
S1 := [m*(P!X[t]) : m in F];

A := [Coefficients(f) : f in S1| f ne 0];
B := [Monomials(f) : f in S1 | f ne 0];
A1:=[];
    for i in [1..#A] do
A1 := A1 cat [[Evaluate(m,X[t],1):m in A[i]]];
    end for;

for i in [1..#A1] do
C[t] := C[t] cat [&+[P!(A1[i][j]*B[i][j]):j in [1..#A1[i]]]];
end for;
end for;
```

```
P2<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,
a15,a14,a13,a12,a11>:=BooleanPolynomialRing(25);

C1:= [P2 ! m : m in C[1]];
C2:= [P2 ! m : m in C[2]];
C3:= [P2 ! m : m in C[3]];
C4:= [P2 ! m : m in C[4]];
C5:= [P2 ! m : m in C[5]];


C := [C1,C2,C3,C4,C5];

return C;

end function;


//////////////////////////////////////////////////////

Boolsudoku := function(F);

G1 := GroebnerBasis(F[1]);
G2 := GroebnerBasis(F[2]);
G3 := GroebnerBasis(F[3]);
G4 := GroebnerBasis(F[4]);
G5 := GroebnerBasis(F[5]);

G11 := [e1*(P!m) : m in G1];
G12 := [e2*(P!m) : m in G2];
G13 := [e3*(P!m) : m in G3];
G14 := [e4*(P!m) : m in G4];
G15 := [e5*(P!m) : m in G5];


G20 := G11 cat G12 cat G13 cat G14 cat G15;

return G20;
end function;

////////////////////////////////////////////

SBGroebner := function(G)

X := {@ LeadingMonomial(m) : m in G @};
n := #X;
Y := [i*(P ! 0) : i in [1..n]];

for i in [1..n] do
    Y[i] := &+[x : x in G | LeadingMonomial(x) eq X[i]];
end for;

return Y;
end function;

////////////////////////////////////////////
```

```
Const := function(F);
Zero := [i*0:i in [1..25]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;




////////////////////////////////////////////////////
////////////////////////////////////////////////////

forward Solpoly, variable,Const, inoue1, inoue11;
inoue1 := function(F: d := 0) /// the main body of Inoue

L := [];
D := [];
I := [0,0,0];
X := [e1,e2,e3,e4,e5];
E := e1+e2+e3+e4+e5;
Zero := [i*0:i in [1..25]];

G:= SBGTrans(F);

S:= Solpoly(G);


d:= d+1;

D := D cat [d];

if #S eq #G then L := L cat [G]; I[1]:= I[1]+1; // one more leaf

end if;

if Const(G) ne 0 then L := L cat [];I[1]:= I[1]+1;// one more leaf

end if;

if #S lt #G and Const(G) eq 0 then //
V := variable(G);
G2:= G;

for s in S do
G2:= [Evaluate(f,LeadingMonomial(s),TrailingTerm(s)) : f in G2];
G2 := [m : m in G2 | m ne 0];
end for;

G3 := [g : g in G2 | IsUnivariate(g,V[1])];

if G3 eq [] then X:= X;
end if;

if G3 ne [] then
```

```
a:= #G3;
zero:= [P!(i*0) : i in [1..a]];
A := [x : x in X | [Evaluate(f,V[1],x): f in G3] eq zero];
if A eq [] then L := L cat []; I[1]:= I[1]+1;// one more leaf

else X:= A;
end if;
end if;

K :=[G cat [E*V[1]+x]: x in X];
K1:= [SBG2(K[i]): i in [1..#K]];
K2:= [m : m in K1 | Const(m) eq 0];
if #K2 eq 0 then L := L cat [];  I[1]:= I[1]+1;

else I[3]:= I[3]+1; // one more node
I[2]:= I[2] + #K2;
end if;


for m in K2 do
L1,I1,D1:= inoue1(m: d:= d);
L := L cat L1;
D := D cat D1;
I[1]:= I[1]+ I1[1];I[2]:= I[2]+I1[2];I[3]:= I[3]+I1[3];
end for;
end if;

return L,I,D;

end function;
//////////////////////////////////////////////////////


Solpoly := function(F); ///output the set of solution polynomials
                        ///contained in the SBG bases F

X := [e1,e2,e3,e4,e5];
E := e1+e2+e3+e4+e5;
Y := [E*a11,E*a12,E*a13,E*a14,E*a15,E*a21,E*a22,E*a23,E*a24,E*a25,
      E*a31,E*a32,E*a33,E*a34,E*a35,E*a41,E*a42,E*a43,E*a44,E*a45,
      E*a51,E*a52,E*a53,E*a54,E*a55];

E := e1+e2+e3+e4+e5;
S := [m : m in F | LeadingTerm(m) in Y and #Terms(m) eq 2 and TrailingTerm(m) in X];
     // the set of solution polynomials

return S;

end function;

////////////////////////////////

variable := function(F);
Y := [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];
```

```
S := Solpoly(F);
K := [LeadingMonomial(m) : m in S];
K1 := [m : m in Y | m notin K];
return K1;

end function;


//////////////////////////////////////////
Const := function(F);
Zero := [i*0:i in [1..25]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;

//////////////////////////////////////////////
//////////////////////////////////////////////


matrix5 := function(F);

Z := IntegerRing();
X1 := [e1,e2,e3,e4,e5];
X2 := [Z!1,Z!2,Z!3,Z!4,Z!5];
W1:=[i*0 : i in [1.25]];

for i in [1..25] do

F[i]:=TrailingTerm(F[i]);

if F[i] eq e1 then W1[i]:=X2[1];
end if;

if F[i] eq e2 then W1[i]:=X2[2];
end if;

if F[i] eq e3 then W1[i]:=X2[3];
end if;

if F[i] eq e4 then W1[i]:=X2[4];
end if;

if F[i] eq e5 then W1[i]:=X2[5];
end if;

end for;

W2 := [Z!m : m in W1];
W3 := [Matrix(5,5,W2)];

return W3;
end function;


//////////////////////////////
//////////////////////////////
```

```
inoue11 := function(F)

F:= SBG2(F);
L,I,D:= inoue1(F);
return L,I,D;
end function;

////////////////////////////////////////////////////////
////////////// Inoue solver for 5-doku ////////////////////

Inoue5 := function(L);
Q3:=[];
I2:=[0,0,0];
Q1,I,D:=inoue11(L);
for i in [1..#Q1] do
Q2:=matrix5(Q1[i]);
Q3:= Q3 cat Q2;
end for;
I[2]:= I[2]+1; I[3]:= I[3]+1;
DM:=Max(D);
I2[1]:=I[1]+I[3]; I2[2]:=I[1]; I2[3]:=DM;

return Q3,I2;

end function;

//////////////////////////////////
```

Computation for Proposition 27

We compute the solutions (and the Inoue invariant) of the diagonal 5-doku puzzle with the initial values $(a_{11}, a_{12}, a_{13}, a_{14}, a_{15}) = (1, 2, 3, 4, 5)$. The 8 solutions obtained are the normalized solution boards in Proposition 27.

OutputLog 6

```
> load "Inoue5.txt";
Loading "C:\Program Files (x86)\Magma\libs\examples\Inoue5.txt"
> load "Input_5.txt";
Loading "C:\Program Files (x86)\Magma\libs\examples\Input_5.txt"
> time Inoue5(G1);
[
    [1 2 3 4 5]
    [2 4 5 3 1]
    [5 3 2 1 4]
    [3 1 4 5 2]
    [4 5 1 2 3],

    [1 2 3 4 5]
    [2 5 4 1 3]
    [4 3 2 5 1]
    [5 4 1 3 2]
    [3 1 5 2 4],
```

```
    [1 2 3 4 5]
    [3 4 5 1 2]
    [5 1 2 3 4]
    [2 3 4 5 1]
    [4 5 1 2 3],

    [1 2 3 4 5]
    [3 5 2 1 4]
    [5 1 4 3 2]
    [4 3 5 2 1]
    [2 4 1 5 3],

    [1 2 3 4 5]
    [4 5 1 2 3]
    [2 3 4 5 1]
    [5 1 2 3 4]
    [3 4 5 1 2],

    [1 2 3 4 5]
    [4 5 2 3 1]
    [5 3 4 1 2]
    [3 1 5 2 4]
    [2 4 1 5 3],

    [1 2 3 4 5]
    [5 3 1 2 4]
    [2 5 4 3 1]
    [4 1 2 5 3]
    [3 4 5 1 2],

    [1 2 3 4 5]
    [5 3 4 1 2]
    [4 5 2 3 1]
    [2 4 1 5 3]
    [3 1 5 2 4]
]
[ 14, 8, 3 ]
Time: 0.390
>
```

The Input_5 file is as follows.

Program 7 (Input_5.txt)

```
G1:=[a11+e1,a12+e2,a13+e3,a14+e4,a15+e5];

////////////////////////////////////////////////////////////////////

E:= e1+e2+e3+e4+e5;
```

```
F:=[
a11+a12+a13+a14+a15+E,a11*a12,a11*a13,a11*a14,a11*a15,a12*a13,a12*a14,a12*a15,a13*a14,a13*a15,a14*a15,
a21+a22+a23+a24+a25+E,a21*a22,a21*a23,a21*a24,a21*a25,a22*a23,a22*a24,a22*a25,a23*a24,a23*a25,a24*a25,
a31+a32+a33+a34+a35+E,a31*a32,a31*a33,a31*a34,a31*a35,a32*a33,a32*a34,a32*a35,a33*a34,a33*a35,a34*a35,
a41+a42+a43+a44+a45+E,a41*a42,a41*a43,a41*a44,a41*a45,a42*a43,a42*a44,a42*a45,a43*a44,a43*a45,a44*a45,
a51+a52+a53+a54+a55+E,a51*a52,a51*a53,a51*a54,a51*a55,a52*a53,a52*a54,a52*a55,a53*a54,a53*a55,a54*a55,

a11+a21+a31+a41+a51+E,a11*a21,a11*a31,a11*a41,a11*a51,a21*a31,a21*a41,a21*a51,a31*a41,a31*a51,a41*a51,
a12+a22+a32+a42+a52+E,a12*a22,a12*a32,a12*a42,a12*a52,a22*a32,a22*a42,a22*a52,a32*a42,a32*a52,a42*a52,
a13+a23+a33+a43+a53+E,a13*a23,a13*a33,a13*a43,a13*a53,a23*a33,a23*a43,a23*a53,a33*a43,a33*a53,a43*a53,
a14+a24+a34+a44+a54+E,a14*a24,a14*a34,a14*a44,a14*a54,a24*a34,a24*a44,a24*a54,a34*a44,a34*a54,a44*a54,
a15+a25+a35+a45+a55+E,a15*a25,a15*a35,a15*a45,a15*a55,a25*a35,a25*a45,a25*a55,a35*a45,a35*a55,a44*a55,

a11^2+a11,a12^2+a12,a13^2+a13,a14^2+a14,a15^2+a15,
a21^2+a21,a22^2+a22,a23^2+a23,a24^2+a24,a25^2+a25,
a31^2+a31,a32^2+a32,a33^2+a33,a34^2+a34,a35^2+a35,
a41^2+a41,a42^2+a42,a43^2+a43,a44^2+a44,a45^2+a45,
a51^2+a51,a52^2+a52,a53^2+a53,a54^2+a54,a55^2+a55,

a11+a22+a33+a44+a55+E,a15+a24+a33+a42+a51+E,

a11*a22,a11*a33,a11*a44,a11*a55,
a22*a33,a22*a44,a22*a55,
a33*a44,a33*a55,
a44*a55,

a15*a24,a15*a33,a15*a42,a15*a51,
a24*a33,a24*a42,a24*a51,
a33*a42,a33*a51,
a42*a51
];


G1:= F cat G1;
```

2.2 Details on the proof of Theorem 28

In the following table, we summarize the number of diagonal 5-doku puzzles with n clues ($3 \leq n \leq 25$) which have the NSB No.1,3,6 as the unique solution.

Table 2: The number of puzzles which have NSB No.1,3,6 as the unique solution

♯(clues)	${}_{25}\mathrm{C}_n$	NSB No.1	NSB No.3	NSB No.6
n=25	1	1	1	1
n=24	25	25	25	25
n=23	300	300	300	300
n=22	2300	2300	2300	2300
n=21	12650	12650	12650	12650
n=20	53130	53130	53130	53130
n=19	177100	177100	177100	177100
n=18	480700	480699	480699	480699
n=17	1081575	1081551	1081551	1081551
n=16	2042975	2042715	2042715	2042715
n=15	3268760	3267032	3267032	3267037
n=14	4457400	4449416	4449416	4449483
n=13	5200300	5172932	5172932	5173326
n=12	5200300	5128105	5128091	5129401
n=11	4457400	4307830	4307649	4310293
n=10	3268760	3023043	3022033	3025227
n=9	2042975	1722599	1719554	1721586
n=8	1081575	909770	907178	907178
n=7	480700	333910	331130	331130
n=6	177100	82474	80926	80926
n=5	53130	10200	9980	9980
n=4	12650	248	244	244
n=3	2300	0	0	0
Total	33554431	32258030	32246636	32256282

From this table, we have

$$(32258030 \times 4 + 32246636 \times 2 + 32256282 \times 2) \times 5! = \mathbf{30964554720}$$

diagonal 5-doku puzzles with a unique solution in all.

The following program computes the number of puzzles with 4 clues which have NSB No.1 as the unique solution. We use the so-called "Naive" algorithm (namely, compute the SBG basis and then determine the value of each variable by substitution starting from the lower variables) for this computation since this is (slightly) faster than the Inoue solver.

Program 8 (Watanabe104.txt)

```
////// basic setting for the coefficient ring //////////////

M2<e1,e2,e3,e4,e5> := PolynomialRing(GF(2),5);
I2:= ideal<M2| e1^2-e1,e2^2-e2,e3^2-e3,e4^2-e4,e5^2-e5,e1*e2,e1*e3,e1*e4,e1*e5,
e2*e3,e2*e4,e2*e5,e3*e4,e3*e5,
```

```
e4*e5>;
M<e1,e2,e3,e4,e5> := M2/I2;

P<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,a15,a14,a13,a12,a11>:=PolynomialRing(M,25);



////////////////////////////////////////////////
////////////// naive ///////////////////////////

forward Seibun, Boolsudoku, SBGroebner;

SBG2:= function(F)

S := Seibun(F);
G := Boolsudoku(S);
G := SBGroebner(G);

return(Sort(G));

end function;

//////////////////////////////////////////////////////////

Seibun:= function(F)

C := [[],[],[],[],[]];
X := [e1,e2,e3,e4,e5];

for t in [1..5] do
S1 := [m*X[t] : m in F];

A := [Coefficients(f) : f in S1| f ne 0];
B := [Monomials(f) : f in S1 | f ne 0];
A1:=[];
    for i in [1..#A] do
A1 := A1 cat [[Evaluate(m,X[t],1):m in A[i]]];
    end for;

for i in [1..#A1] do
C[t] := C[t] cat [&+[P!(A1[i][j]*B[i][j]):j in [1..#A1[i]]]];
end for;
end for;

P2<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,
a15,a14,a13,a12,a11>:=BooleanPolynomialRing(25);

C1:= [P2 ! m : m in C[1]];
C2:= [P2 ! m : m in C[2]];
C3:= [P2 ! m : m in C[3]];
C4:= [P2 ! m : m in C[4]];
C5:= [P2 ! m : m in C[5]];


C := [C1,C2,C3,C4,C5];
```

```
return C;

end function;


//////////////////////////////////////////////////////

Boolsudoku := function(F);

G1 := GroebnerBasis(F[1]);
G2 := GroebnerBasis(F[2]);
G3 := GroebnerBasis(F[3]);
G4 := GroebnerBasis(F[4]);
G5 := GroebnerBasis(F[5]);



G11 := [e1*(P!m) : m in G1];
G12 := [e2*(P!m) : m in G2];
G13 := [e3*(P!m) : m in G3];
G14 := [e4*(P!m) : m in G4];
G15 := [e5*(P!m) : m in G5];


G20 := G11 cat G12 cat G13 cat G14 cat G15;

return G20;
end function;

////////////////////////////////////////////

SBGroebner := function(G)

X := {@ LeadingMonomial(m) : m in G @};
n := #X;
Y := [i*(P ! 0) : i in [1..n]];

for i in [1..n] do
    Y[i] := &+[x : x in G | LeadingMonomial(x) eq X[i]];
end for;

return Y;
end function;

////////////////////////////////////////////

Const := function(F);
Zero := [i*0:i in [1..25]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;

//////////////////////////////////////////////////////
//////////////////////////////////////////////////////

forward Dainyuu;
```

```
Backsub := function(F)

t := 0;
Y := [m*0 : m in [1..25]]; // a sequence in which a solution is put

W := Dainyuu(F,Y,t);
// #W;
return W;
end function;

////////////////////////////////////////////

Dainyuu := function(F,Y,t)

X := [e1,e2,e3,e4,e5];
G1 := F;
s := t;
W:=[];

Z := [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];

for t in [s+1..#Z] do

// find an element in G1 such that only a11 is containd etc.
A := [m : m in G1 | Degree(m,Z[t]) gt 0];
for i in [t+1..#Z] do
A := [m: m in A | Degree(m,Z[i]) eq 0];
end for;

check := 0;
////seek for a solution
for i in [1..5] do
A1 := [Evaluate(m,Z[t],X[i]) : m in A];
H := [m :m in A1 | m ne 0];
if #H eq 0 then
if check eq 0 then //branch check
Y[t] := i;
check := 1;
else //when branched, compute
G2 := [Evaluate(m,Z[t],X[Y[t]]) : m in G1];
G2 := [m : m in G2 | m ne 0];
Z2 := Dainyuu(G2,Y,t);

W := W cat Z2;
                     if #W ge 2 then return W;
                    end if;

Y[t] := i;
end if;
end if;
end for;

////substitution
if Y[t] eq 0 then
return W;
else
```

```
G1 := [Evaluate(m,Z[t],X[Y[t]]) : m in G1];
G1 := [m : m in G1 | m ne 0];
end if;
end for;
Z3 := [Matrix(5,5,Y)];
W := W cat Z3;
if #W ge 2 then return W;
end if;

return W;
end function;

////////////////////////////////////////////////////////
///////////////// This is the naive ////////////////////////////

naive:= function(F)

G := Backsub(SBG2(F));

return G;

end function;


//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////

L:= [a11+e1,a12+e2,a13+e3,a14+e4,a15+e5,a21+e2,a22+e4,a23+e5,a24+e3,a25+e1,a31+e5,a32+e3,a33+e2,
a34+e1,a35+e4,a41+e3,a42+e1,a43+e4,a44+e5,a45+e2,a51+e4,a52+e5,a53+e1,a54+e2,a55+e3];
///// initial values of the solution board NSB No.1
///// modify this line in the other cases

F:=[a11+a21+a31+a41+a51+1,a12+a22+a32+a42+a52+1,a13+a23+a33+a43+a53+1,
a14+a24+a34+a44+a54+1,a15+a25+a35+a45+a55+1,

a11+a12+a13+a14+a15+1,a21+a22+a23+a24+a25+1,a31+a32+a33+a34+a35+1,
a41+a42+a43+a44+a45+1,a51+a52+a53+a54+a55+1,

a11+a22+a33+a44+a55+1,a15+a24+a33+a42+a51+1,

a11*a22,a11*a33,a11*a44,a11*a55,
a22*a33,a22*a44,a22*a55,
a33*a44,a33*a55,
a44*a55,

a15*a24,a15*a33,a15*a42,a15*a51,
a24*a33,a24*a42,a24*a51,
a33*a42,a33*a51,
a42*a51,



a11*a12,a11*a13,a11*a14,a11*a15,
a12*a13,a12*a14,a12*a15,
a13*a14,a13*a15,
a14*a15,
```

```
a21*a22,a21*a23,a21*a24,a21*a25,
a22*a23,a22*a24,a22*a25,
a23*a24,a23*a25,
a24*a25,


a31*a32,a31*a33,a31*a34,a31*a35,
a32*a33,a32*a34,a32*a35,
a33*a34,a33*a35,
a34*a35,


a41*a42,a41*a43,a41*a44,a41*a45,
a42*a43,a42*a44,a42*a45,
a43*a44,a43*a45,
a44*a45,


a51*a52,a51*a53,a51*a54,a51*a55,
a52*a53,a52*a54,a52*a55,
a53*a54,a53*a55,
a54*a55,


a11*a21,a11*a31,a11*a41,a11*a51,
a21*a31,a21*a41,a21*a51,
a31*a41,a31*a51,
a41*a51,

a12*a22,a12*a32,a12*a42,a12*a52,
a22*a32,a22*a42,a22*a52,
a32*a42,a32*a52,
a42*a52,


a13*a23,a13*a33,a13*a43,a13*a53,
a23*a33,a23*a43,a23*a53,
a33*a43,a33*a53,
a43*a53,

a14*a24,a14*a34,a14*a44,a14*a54,
a24*a34,a24*a44,a24*a54,
a34*a44,a34*a54,
a44*a54,

a15*a25,a15*a35,a15*a45,a15*a55,
a25*a35,a25*a45,a25*a55,
a35*a45,a35*a55,
a45*a55,



a11^2+a11,a12^2+a12,a13^2+a13,a14^2+a14,a15^2+a15,
a21^2+a21,a22^2+a22,a23^2+a23,a24^2+a24,a25^2+a25,
a31^2+a31,a32^2+a32,a33^2+a33,a34^2+a34,a35^2+a35,
a41^2+a41,a42^2+a42,a43^2+a43,a44^2+a44,a45^2+a45,
```

```
a51^2+a51,a52^2+a52,a53^2+a53,a54^2+a54,a55^2+a55

];

////////////////////////////////////////////////
//////// Watanabe104 ////////////////////////

Watanabe104 := procedure();

count := 0;

///// modify for part below in the other cases
for x4 in [1..22] do
for x3 in [x4+1..23] do
for x2 in [x3+1..24] do
for x1 in [x2+1..25] do

Z:= [L[x4],L[x3],L[x2],L[x1]];
I := F cat Z;

if #naive(I) eq 1 then
count:= count +1;
end if;

end for;
end for;
end for;
end for;

count;

end procedure;
```

The output log is as follows.

OutputLog 9

```
> load "Watanabe104.txt";
Loading "C:\Program Files (x86)\Magma\libs\examples\Watanabe104.txt"
> time Watanabe104();
248
Time: 933.058
>
```

Thus we find that there are 248 puzzles with 4 clues which have the NSB No.1 as the unique solution. Similar programs with k clues ($3 \leq k \leq 25$) for the NSB No.1,3,6 give the data in Table 2 (we need to modify the last part of Program 8 in two places).

We next turn to the irredundant puzzles. The following table gives the number of irredundant diagonal 5-doku puzzles with n clues ($3 \leq n \leq 25$) which have NSB No.1,3,6 as the unique solution.

Table 3: The number of irredundant puzzles which have NSB No.1,3,6 as the unique solution

♯(clues)	${}_{25}\mathrm{C}_n$	NSB No.1	NSB No.3	NSB No.6
n=25	1	0	0	0
n=24	25	0	0	0
n=23	300	0	0	0
n=22	2300	0	0	0
n=21	12650	0	0	0
n=20	53130	0	0	0
n=19	177100	0	0	0
n=18	480700	0	0	0
n=17	1081575	0	0	0
n=16	2042975	0	0	0
n=15	3268760	0	0	0
n=14	4457400	0	0	0
n=13	5200300	0	0	0
n=12	5200300	0	0	0
n=11	4457400	0	0	0
n=10	3268760	0	0	0
n=9	2042975	0	0	0
n=8	1081575	0	0	0
n=7	480700	0	0	0
n=6	177100	2144	2200	2200
n=5	53130	5604	5476	5476
n=4	12650	248	244	244
n=3	2300	0	0	0
Total	33554431	7996	7920	7920

From this table, we know that the irredundant puzzles exist only when the number of clues is $4, 5, 6$, and we have

$$(7996 \times 4 + 7920 \times 2 + 7920 \times 2) \times 5! = \mathbf{7639680}$$

irredundant puzzles in all.

The following program computes the number of irredundant puzzles with 5 clues which have the NSB No.1 as the unique solution.

Program 10 (WatanaebM105)

```
M2<e1,e2,e3,e4,e5> := PolynomialRing(GF(2),5);
I2:= ideal<M2| e1^2-e1,e2^2-e2,e3^2-e3,e4^2-e4,e5^2-e5,e1*e2,e1*e3,e1*e4,e1*e5,
e2*e3,e2*e4,e2*e5,e3*e4,e3*e5,
e4*e5>;
M<e1,e2,e3,e4,e5> := M2/I2;

P<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
```

```
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,a15,a14,a13,a12,a11>:=PolynomialRing(M,25);



//////////////////////////// naive ////////////////////////////////////////////////

forward Seibun, Boolsudoku, SBGroebner;

SBG2:= function(F)

S := Seibun(F);
G := Boolsudoku(S);
G := SBGroebner(G);

return(Sort(G));

end function;

//////////////////////////////////////////////////////////

Seibun:= function(F)

C := [[],[],[],[],[]];
X := [e1,e2,e3,e4,e5];

for t in [1..5] do
S1 := [m*X[t] : m in F];

A := [Coefficients(f) : f in S1| f ne 0];
B := [Monomials(f) : f in S1 | f ne 0];
A1:=[];
    for i in [1..#A] do
A1 := A1 cat [[Evaluate(m,X[t],1):m in A[i]]];
    end for;

for i in [1..#A1] do
C[t] := C[t] cat [&+[P!(A1[i][j]*B[i][j]):j in [1..#A1[i]]]];
end for;
end for;

P2<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,
a15,a14,a13,a12,a11>:=BooleanPolynomialRing(25);

C1:= [P2 ! m : m in C[1]];
C2:= [P2 ! m : m in C[2]];
C3:= [P2 ! m : m in C[3]];
C4:= [P2 ! m : m in C[4]];
C5:= [P2 ! m : m in C[5]];


C := [C1,C2,C3,C4,C5];

return C;

end function;
```

```
//////////////////////////////////////////////////////

Boolsudoku := function(F);

G1 := GroebnerBasis(F[1]);
G2 := GroebnerBasis(F[2]);
G3 := GroebnerBasis(F[3]);
G4 := GroebnerBasis(F[4]);
G5 := GroebnerBasis(F[5]);



G11 := [e1*(P!m) : m in G1];
G12 := [e2*(P!m) : m in G2];
G13 := [e3*(P!m) : m in G3];
G14 := [e4*(P!m) : m in G4];
G15 := [e5*(P!m) : m in G5];


G20 := G11 cat G12 cat G13 cat G14 cat G15;

return G20;
end function;

////////////////////////////////////////////

SBGroebner := function(G)

X := {@ LeadingMonomial(m) : m in G @};
n := #X;
Y := [i*(P ! 0) : i in [1..n]];

for i in [1..n] do
    Y[i] := &+[x : x in G | LeadingMonomial(x) eq X[i]];
end for;

return Y;
end function;

////////////////////////////////////////////

Const := function(F);
Zero := [i*0:i in [1..25]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;

//////////////////////////////////////////////////////
//////////////////////////////////////////////////////

forward Dainyuu;

Backsub := function(F)

t := 0;
```

```
Y := [m*0 : m in [1..25]]; // a sequence in which a solution is put

W := Dainyuu(F,Y,t);
// #W;
return W;
end function;

////////////////////////////////////////////

Dainyuu := function(F,Y,t)

X := [e1,e2,e3,e4,e5];
G1 := F;
s := t;
W:=[];

Z := [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];

for t in [s+1..#Z] do

// find an element in G1 such that only a11 is containd etc.
A := [m : m in G1 | Degree(m,Z[t]) gt 0];
for i in [t+1..#Z] do
A := [m: m in A | Degree(m,Z[i]) eq 0];
end for;

check := 0;
//seek for a solution
for i in [1..5] do
A1 := [Evaluate(m,Z[t],X[i]) : m in A];
H := [m :m in A1 | m ne 0];
if #H eq 0 then
if check eq 0 then //branch check
Y[t] := i;
check := 1;
else //when branched, compute
G2 := [Evaluate(m,Z[t],X[Y[t]]) : m in G1];
G2 := [m : m in G2 | m ne 0];
Z2 := Dainyuu(G2,Y,t);

W := W cat Z2;
                      if #W ge 2 then return W;
                      end if;

Y[t] := i;
end if;
end if;
end for;

//substitution
if Y[t] eq 0 then
return W;
else
G1 := [Evaluate(m,Z[t],X[Y[t]]) : m in G1];
G1 := [m : m in G1 | m ne 0];
end if;
end for;
```

```
Z3 := [Matrix(5,5,Y)];
W := W cat Z3;
if #W ge 2 then return W;
end if;

return W;
end function;
////////////////////////////////////////////////////////////


naive:= function(F)

G := Backsub(SBG2(F));

return G;


end function;


//////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////

L:= [a11+e1,a12+e2,a13+e3,a14+e4,a15+e5,a21+e2,a22+e4,a23+e5,a24+e3,a25+e1,a31+e5,a32+e3,a33+e2,
a34+e1,a35+e4,a41+e3,a42+e1,a43+e4,a44+e5,a45+e2,a51+e4,a52+e5,a53+e1,a54+e2,a55+e3];
////// modify this line in the other cases

F:=[a11+a21+a31+a41+a51+1,a12+a22+a32+a42+a52+1,a13+a23+a33+a43+a53+1,
a14+a24+a34+a44+a54+1,a15+a25+a35+a45+a55+1,

a11+a12+a13+a14+a15+1,a21+a22+a23+a24+a25+1,a31+a32+a33+a34+a35+1,
a41+a42+a43+a44+a45+1,a51+a52+a53+a54+a55+1,

a11+a22+a33+a44+a55+1,a15+a24+a33+a42+a51+1,

a11*a22,a11*a33,a11*a44,a11*a55,
a22*a33,a22*a44,a22*a55,
a33*a44,a33*a55,
a44*a55,

a15*a24,a15*a33,a15*a42,a15*a51,
a24*a33,a24*a42,a24*a51,
a33*a42,a33*a51,
a42*a51,



a11*a12,a11*a13,a11*a14,a11*a15,
a12*a13,a12*a14,a12*a15,
a13*a14,a13*a15,
a14*a15,


a21*a22,a21*a23,a21*a24,a21*a25,
a22*a23,a22*a24,a22*a25,
a23*a24,a23*a25,
a24*a25,
```

```
a31*a32,a31*a33,a31*a34,a31*a35,
a32*a33,a32*a34,a32*a35,
a33*a34,a33*a35,
a34*a35,


a41*a42,a41*a43,a41*a44,a41*a45,
a42*a43,a42*a44,a42*a45,
a43*a44,a43*a45,
a44*a45,


a51*a52,a51*a53,a51*a54,a51*a55,
a52*a53,a52*a54,a52*a55,
a53*a54,a53*a55,
a54*a55,


a11*a21,a11*a31,a11*a41,a11*a51,
a21*a31,a21*a41,a21*a51,
a31*a41,a31*a51,
a41*a51,

a12*a22,a12*a32,a12*a42,a12*a52,
a22*a32,a22*a42,a22*a52,
a32*a42,a32*a52,
a42*a52,


a13*a23,a13*a33,a13*a43,a13*a53,
a23*a33,a23*a43,a23*a53,
a33*a43,a33*a53,
a43*a53,

a14*a24,a14*a34,a14*a44,a14*a54,
a24*a34,a24*a44,a24*a54,
a34*a44,a34*a54,
a44*a54,

a15*a25,a15*a35,a15*a45,a15*a55,
a25*a35,a25*a45,a25*a55,
a35*a45,a35*a55,
a45*a55,



a11^2+a11,a12^2+a12,a13^2+a13,a14^2+a14,a15^2+a15,
a21^2+a21,a22^2+a22,a23^2+a23,a24^2+a24,a25^2+a25,
a31^2+a31,a32^2+a32,a33^2+a33,a34^2+a34,a35^2+a35,
a41^2+a41,a42^2+a42,a43^2+a43,a44^2+a44,a45^2+a45,
a51^2+a51,a52^2+a52,a53^2+a53,a54^2+a54,a55^2+a55

];

//////////////////////////////////////////////////////
```

```
/////////  WatanabeM105  ////////////////////////////

WatanabeM105 := procedure();

count := 0;

///// modify for part below in the other cases
for x5 in [1..21] do
for x4 in [x5+1..22] do
for x3 in [x4+1..23] do
for x2 in [x3+1..24] do
for x1 in [x2+1..25] do


Z:= [L[x5],
L[x4],L[x3],L[x2],L[x1]];
I := F cat Z;
if #naive(I) eq 1 then
count1 :=0;

for i in [1..5] do
T:= Remove(Z,i);
if #naive(F cat T) ge 2 then count1 := count1+ 1;
end if;
end for;

if count1 eq 5 then
count := count +1;
Z;
end if;

end if;

end for;
end for;
end for;
end for;
end for;


print "MudaNoNaiHaichiSuu=", count;

end procedure;
```

The output log is like this.

OutputLog 11

```
> load "WatanabeM105.txt";
> time WatanabeM105();

/// omitted since lengthy

[
```

```
    a44 + e5,
    a45 + e2,
    a51 + e4,
    a54 + e2,
    a55 + e3
]
[
    a44 + e5,
    a45 + e2,
    a52 + e5,
    a53 + e1,
    a55 + e3
]
MudaNoNaiHaichiSuu= 5604
Time: 7489.062 (seconds)
>
```

Thus we find that there are 5604 irredundant puzzles with 5 clues which have the NSB No.1 as the unique solution. The other cases can be computed similarly (we need to modify the last part pf Program 10) and we have Table 3.

2.3 Programs for Theorem 31

The following program computes the Inoue invariant of the puzzles with 6 clues which have the NSB No.1 as the unique solution.

Program 12 (Inoue516.txt)

```
M2<e1,e2,e3,e4,e5> := PolynomialRing(GF(2),5);
I2:= ideal<M2| e1^2-e1,e2^2-e2,e3^2-e3,e4^2-e4,e5^2-e5,e1*e2,e1*e3,e1*e4,e1*e5,
e2*e3,e2*e4,e2*e5,e3*e4,e3*e5,
e4*e5>;
M<e1,e2,e3,e4,e5> := M2/I2;

P<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,a15,a14,a13,a12,a11>:=PolynomialRing(M,25);


////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////

forward AlmostSolution3, SBG2, Const;

SBGTrans := function(F)

if Const(F) ne 0 then return F;
end if;

G:= F;
check := 0;
n:= [1,1,1,1];

while n[2]+ n[3]+n[4] ne 0 and check eq 0 do
```

```
n,G := AlmostSolution3(G);

G:= SBG2(G);

if Const(G) ne 0 then check := 1;
end if;

end while;

return G;

end function;

////////////////////////////////////////////////////



AlmostSolution3 := function(F);

X := [e1,e2,e3,e4,e5];
Y := [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];
E := e1+e2+e3+e4+e5;
Y1:= [E*y: y in Y];

G := F;
Zero := [i*0:i in [1..25]];
Z1 := [E*m2+m1 : m1 in X,m2 in Y];
Z2 := [(E+m1)*m2:m1 in X,m2 in Y];

S := [m : m in G | m in Z1]; // the set of solution polynomials
S1 := [m : m in G | m in Z2];// the set of AS polynomial of type 1
S3 := S cat S1;

W := SequenceToSet(G) diff SequenceToSet(S3);
G := SetToSequence(W); ///the set of SBG bases with solution polynomials and
                       /// ASP's of type 1 removed


G0:=[ g: g in G | LeadingTerm(g) in Y1];
S0:= [];
T0:= [];

for g in G0 do
U0:= [e: e in X | e*(g-LeadingTerm(g)) eq g - LeadingTerm(g)];
if #U0 eq 1 then S0:= S0 cat [g]; T0 := T0 cat [LeadingTerm(g)+U0[1]];
end if;
end for;



T2 := [];
S2:= [];
```

```
for i in [1..#G] do
check := 0;
        for j in [1..5] do
if X[j]*G[i] eq X[j]*LeadingTerm(G[i])+X[j] and LeadingMonomial(G[i]) in Y then
                        check := check +1;
T2 := T2 cat [E*LeadingMonomial(G[i])+X[j]];
end if;
end for;
if check ne 0 then S2:= S2 cat [G[i]]; end if;
end for;

T1 := [E*LeadingMonomial(m) + (E + LeadingCoefficient(m)) : m in S1];
      //// transform AS polynomials of type 1 to solution polynomials
n := [#S,#S2,#S1,#S0];
S3 :=S2 cat S1;

G := F cat T1 cat T2 cat T0; /// add T1 and T2 to F, not F-(S1 +S2)

return n, G;

end function;

////////////////////////////////////////////////////////////////////
forward Seibun, Boolsudoku, SBGroebner;

SBG2:= function(F)

S := Seibun(F);
G := Boolsudoku(S);
G := SBGroebner(G);

return(Sort(G));

end function;

//////////////////////////////////////////////////////////

Seibun:= function(F)

C := [[],[],[],[],[]];
X := [e1,e2,e3,e4,e5];

for t in [1..5] do
S1 := [m*(P!X[t]) : m in F];

A := [Coefficients(f) : f in S1| f ne 0];
B := [Monomials(f) : f in S1 | f ne 0];
A1:=[];
    for i in [1..#A] do
A1 := A1 cat [[Evaluate(m,X[t],1):m in A[i]]];
    end for;

for i in [1..#A1] do
C[t] := C[t] cat [&+[P!(A1[i][j]*B[i][j]):j in [1..#A1[i]]]];
end for;
end for;
```

```
P2<a55,a54,a53,a52,a51,a45,a44,a43,a42,a41,
a35,a34,a33,a32,a31,a25,a24,a23,a22,a21,
a15,a14,a13,a12,a11>:=BooleanPolynomialRing(25);

C1:= [P2 ! m : m in C[1]];
C2:= [P2 ! m : m in C[2]];
C3:= [P2 ! m : m in C[3]];
C4:= [P2 ! m : m in C[4]];
C5:= [P2 ! m : m in C[5]];


C := [C1,C2,C3,C4,C5];

return C;

end function;


//////////////////////////////////////////////////////

Boolsudoku := function(F);

G1 := GroebnerBasis(F[1]);
G2 := GroebnerBasis(F[2]);
G3 := GroebnerBasis(F[3]);
G4 := GroebnerBasis(F[4]);
G5 := GroebnerBasis(F[5]);

G11 := [e1*(P!m) : m in G1];
G12 := [e2*(P!m) : m in G2];
G13 := [e3*(P!m) : m in G3];
G14 := [e4*(P!m) : m in G4];
G15 := [e5*(P!m) : m in G5];


G20 := G11 cat G12 cat G13 cat G14 cat G15;

return G20;
end function;

////////////////////////////////////////////

SBGroebner := function(G)

X := {@ LeadingMonomial(m) : m in G @};
n := #X;
Y := [i*(P ! 0) : i in [1..n]];

for i in [1..n] do
    Y[i] := &+[x : x in G | LeadingMonomial(x) eq X[i]];
end for;

return Y;
end function;

////////////////////////////////////////////
```

```
Const := function(F);
Zero := [i*0:i in [1..25]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;


////////////////////////////////////////////////
////////////////////////////////////////////////

forward Solpoly, variable,Const, inoue1, inoue11;
inoue1 := function(F: d := 0) /// the main body of Inoue

L := [];
D := [];
I := [0,0,0];
X := [e1,e2,e3,e4,e5];
E := e1+e2+e3+e4+e5;
Zero := [i*0:i in [1..25]];

G:= SBGTrans(F);

S:= Solpoly(G);


d:= d+1;

D := D cat [d];

if #S eq #G then L := L cat [G]; I[1]:= I[1]+1; // one more leaf

end if;

if Const(G) ne 0 then L := L cat [];I[1]:= I[1]+1;// one more leaf

end if;

if #S lt #G and Const(G) eq 0 then //
V := variable(G);
G2:= G;

for s in S do
G2:= [Evaluate(f,LeadingMonomial(s),TrailingTerm(s)) : f in G2];
G2 := [m : m in G2 | m ne 0];
end for;

G3 := [g : g in G2 | IsUnivariate(g,V[1])];

if G3 eq [] then X:= X;
end if;

if G3 ne [] then
a:= #G3;
zero:= [P!(i*0) : i in [1..a]];
A := [x : x in X | [Evaluate(f,V[1],x): f in G3] eq zero];
if A eq [] then L := L cat []; I[1]:= I[1]+1;// one more leaf
```

```
else X:= A;
end if;
end if;

K :=[G cat [E*V[1]+x]: x in X];
K1:= [SBG2(K[i]): i in [1..#K]];
K2:= [m : m in K1 | Const(m) eq 0];
if #K2 eq 0 then L := L cat [];  I[1]:= I[1]+1;

else I[3]:= I[3]+1; // one more node
I[2]:= I[2] + #K2;
end if;


for m in K2 do
L1,I1,D1:= inoue1(m: d:= d);
L := L cat L1;
D := D cat D1;
I[1]:= I[1]+ I1[1];I[2]:= I[2]+I1[2];I[3]:= I[3]+I1[3];
end for;
end if; //

return L,I,D;

end function;
////////////////////////////////////////////////////


Solpoly := function(F); ///output the set of solution polynomials
                        ///contained in the SBG bases F

X := [e1,e2,e3,e4,e5];
E := e1+e2+e3+e4+e5;
Y := [E*a11,E*a12,E*a13,E*a14,E*a15,E*a21,E*a22,E*a23,E*a24,E*a25,
      E*a31,E*a32,E*a33,E*a34,E*a35,E*a41,E*a42,E*a43,E*a44,E*a45,
      E*a51,E*a52,E*a53,E*a54,E*a55];

E := e1+e2+e3+e4+e5;
S := [m : m in F | LeadingTerm(m) in Y and #Terms(m) eq 2 and TrailingTerm(m) in X];
     // the set of solution polynomials

return S;

end function;

////////////////////////////////

variable := function(F);
Y := [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];

S := Solpoly(F);
K := [LeadingMonomial(m) : m in S];
K1 := [m : m in Y | m notin K];
return K1;
```

```
end function;


//////////////////////////////////////////
Const := function(F);
Zero := [i*0:i in [1..25]];
U := [m: m in F | Exponents(LeadingTerm(m)) eq Zero];
return #U;
end function;

//////////////////////////////////////////
//////////////////////////////////////////


matrix5 := function(F);

Z := IntegerRing();
X1 := [e1,e2,e3,e4,e5];
X2 := [Z!1,Z!2,Z!3,Z!4,Z!5];
W1:=[i*0 : i in [1.25]];

for i in [1..25] do

F[i]:=TrailingTerm(F[i]);

if F[i] eq e1 then W1[i]:=X2[1];
end if;

if F[i] eq e2 then W1[i]:=X2[2];
end if;

if F[i] eq e3 then W1[i]:=X2[3];
end if;

if F[i] eq e4 then W1[i]:=X2[4];
end if;

if F[i] eq e5 then W1[i]:=X2[5];
end if;

end for;

W2 := [Z!m : m in W1];
W3 := [Matrix(5,5,W2)];

return W3;
end function;


//////////////////////////////
//////////////////////////////

inoue11 := function(F)

F:= SBG2(F);
L,I,D:= inoue1(F);
return L,I,D;
```

```
end function;


//////////////////////////
////////////////////////////

Inoue5 := function(L);
Q3:=[];
I2:=[0,0,0];
Q1,I,D:=inoue11(L);
for i in [1..#Q1] do
Q2:=matrix5(Q1[i]);
Q3:= Q3 cat Q2;
end for;
I[2]:= I[2]+1; I[3]:= I[3]+1;
DM:=Max(D);
I2[1]:=I[1]+I[3]; I2[2]:=I[1]; I2[3]:=DM;

return Q3,I2;

end function;

////////////////////////////////

L:=[a11+e1,a12+e2,a13+e3,a14+e4,a15+e5,a21+e2,a22+e4,a23+e5,a24+e3,a25+e1,
a31+e5,a32+e3,a33+e2,a34+e1,a35+e4,a41+e3,a42+e1,a43+e4,a44+e5,a45+e2,a51+e4,a52+e5,a53+e1,a54+e2,a55+e3];
///// modify this line in the other cases


E:= e1+e2+e3+e4+e5;


F:=[
a11+a12+a13+a14+a15+E,a11*a12,a11*a13,a11*a14,a11*a15,a12*a13,a12*a14,a12*a15,a13*a14,a13*a15,a14*a15,
a21+a22+a23+a24+a25+E,a21*a22,a21*a23,a21*a24,a21*a25,a22*a23,a22*a24,a22*a25,a23*a24,a23*a25,a24*a25,
a31+a32+a33+a34+a35+E,a31*a32,a31*a33,a31*a34,a31*a35,a32*a33,a32*a34,a32*a35,a33*a34,a33*a35,a34*a35,
a41+a42+a43+a44+a45+E,a41*a42,a41*a43,a41*a44,a41*a45,a42*a43,a42*a44,a42*a45,a43*a44,a43*a45,a44*a45,
a51+a52+a53+a54+a55+E,a51*a52,a51*a53,a51*a54,a51*a55,a52*a53,a52*a54,a52*a55,a53*a54,a53*a55,a54*a55,

a11+a21+a31+a41+a51+E,a11*a21,a11*a31,a11*a41,a11*a51,a21*a31,a21*a41,a21*a51,a31*a41,a31*a51,a41*a51,
a12+a22+a32+a42+a52+E,a12*a22,a12*a32,a12*a42,a12*a52,a22*a32,a22*a42,a22*a52,a32*a42,a32*a52,a42*a52,
a13+a23+a33+a43+a53+E,a13*a23,a13*a33,a13*a43,a13*a53,a23*a33,a23*a43,a23*a53,a33*a43,a33*a53,a43*a53,
a14+a24+a34+a44+a54+E,a14*a24,a14*a34,a14*a44,a14*a54,a24*a34,a24*a44,a24*a54,a34*a44,a34*a54,a44*a54,
a15+a25+a35+a45+a55+E,a15*a25,a15*a35,a15*a45,a15*a55,a25*a35,a25*a45,a25*a55,a35*a45,a35*a55,a44*a55,

a11^2+a11,a12^2+a12,a13^2+a13,a14^2+a14,a15^2+a15,
a21^2+a21,a22^2+a22,a23^2+a23,a24^2+a24,a25^2+a25,
a31^2+a31,a32^2+a32,a33^2+a33,a34^2+a34,a35^2+a35,
a41^2+a41,a42^2+a42,a43^2+a43,a44^2+a44,a45^2+a45,
a51^2+a51,a52^2+a52,a53^2+a53,a54^2+a54,a55^2+a55,

a11+a22+a33+a44+a55+E,a15+a24+a33+a42+a51+E,

a11*a22,a11*a33,a11*a44,a11*a55,
a22*a33,a22*a44,a22*a55,
a33*a44,a33*a55,
a44*a55,
```

```
a15*a24,a15*a33,a15*a42,a15*a51,
a24*a33,a24*a42,a24*a51,
a33*a42,a33*a51,
a42*a51
];

////////////////////////////////////////////
////////// Inoue516 ////////////////////////

Inoue516 := procedure();

count1 := 0;
count2 := 0;

///// modify for part below in the other cases
for x6 in [1..20] do
for x5 in [x6+1..21] do
for x4 in [x5+1..22] do
for x3 in [x4+1..23] do
for x2 in [x3+1..24] do
for x1 in [x2+1..25] do

Z:= [L[x6],L[x5],L[x4],L[x3],L[x2],L[x1]];
I := F cat Z;

S3,I2 := Inoue5(I);

if #S3 eq 1 then
count1 :=count1 +1;
end if;

if #S3 eq 1 and I2 eq [2,1,1] then
count2 := count2 +1;
end if;

if #S3 eq 1 and I2 ne [2,1,1] then
Z; I2;
end if;

end for;
end for;
end for;
end for;
end for;
end for;

"the number of puzzles with a unique solution =",count1;
"the number of puzzles with a trivial Inoue invariant =",count2;

end procedure;
```

The output log is as follows.

OutputLog 13

```
Magma V2.18-3     Mon Mar 11 2013 17:45:33 on vostro   [Seed = 2296660053]
Type ? for help.  Type <Ctrl>-D to quit.
> load "Inoue516.txt";
Loading "C:\Program Files (x86)\Magma\libs\examples\Inoue516.txt"
> time Inoue516();
[
    a11 + e1,
    a14 + e4,
    a15 + e5,
    a41 + e3,
    a42 + e1,
    a44 + e5
]
[ 4, 2, 2 ]
[
    a11 + e1,
    a22 + e4,
    a41 + e3,
    a42 + e1,
    a51 + e4,
    a52 + e5
]
[ 4, 2, 2 ]
[
    a14 + e4,
    a15 + e5,
    a24 + e3,
    a25 + e1,
    a44 + e5,
    a55 + e3
]
[ 4, 2, 2 ]
[
    a22 + e4,
    a24 + e3,
    a25 + e1,
    a51 + e4,
    a52 + e5,
    a55 + e3
]
[ 4, 2, 2 ]
the number of puzzles with a unique solution = 82474
the number of puzzles with a trivial Inoue invariant = 82470
Time: 19106.909
>
```

So we find that there are 4 puzzles with 6 clues with the non-trivial Inoue invariant $(4, 2, 2)$ which have the NSB No.1 as the unique solution. We note that the first and the fourth puzzles (resp. the second and the third ones) are transformed each other by the $S_5 \times D_4$-action.

Similar programs with n clues ($4 \leq n \leq 6$) for the NSB No.1, No. 3 and No.6 show that these two are the only puzzles with non-trivial Inoue invariants (we need to modify the last part of Program 12).