Abominable TRIAL-and-ERROR and lovely BRAIDS

Goto page Previous 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 Next

newtopic postreply Sudoku Players' Forums Forum Index -> Advanced solving techniques

Author	Message

denis_berthier

Posted: Wed Nov 25, 2009 6:02 am Post subject:

quote

Joined: 19 Jun 2007
Posts: 1123
Location: Paris, France

Here is another braids example, this time with an unusually large number (6) of consecutive braids (usually, there are only a few braids lost amidst lots of whips)
There is also an unusual number of chain patterns leading to several eliminations.
Only braids are activated (we get the pB-NRCZT rating)
I've manually edited the final cell of non-whip braids so that it looks better.

Code:

```
______________________________________________________________________________________________________________________
*****  SudoRules version 13.7wbisB2  *****
....5.....567....28..23..6......78.....3...2...1.2.3.734.9...1...21...7....8...9. 306431
hidden-single-in-row r2 ==> r2c8 = 3
hidden-single-in-column c8 ==> r1c8 = 8
interaction column c8 with block b6 for number 5 ==> r5c9 <> 5, r5c7 <> 5, r4c9 <> 5
interaction column c8 with block b6 for number 4 ==> r5c9 <> 4, r5c7 <> 4, r4c9 <> 4
interaction column c4 with block b5 for number 5 ==> r6c6 <> 5, r5c6 <> 5
interaction row r5 with block b4 for number 5 ==> r6c1 <> 5, r4c3 <> 5, r4c1 <> 5
nrc-chain[2]  n2{r1c2 r4c2} - n3{r4c2 r1c2} ==> r1c2 <> 9, r1c2 <> 7, r1c2 <> 1
nrc-chain[2]  n2{r4c2 r1c2} - n3{r1c2 r4c2} ==> r4c2 <> 9, r4c2 <> 6
xyz-chain[3]  {n6 n7}r7c5 - {n7 n4}r9c5 - {n4 n6}r8c5 ==> r9c6 <> 6, r8c6 <> 6, r7c6 <> 6
interaction block b8 with column c5 for number 6 ==> r5c5 <> 6, r4c5 <> 6
nrct-chain[3]  n3{r8c6 r9c6} - n2{r9c6 r7c6} - n5{r7c6 r8c6} ==> r8c6 <> 4
nrct-chain[3]  {n4 n6}r8c5 - {n6 n7}r7c5 - {n7 n4}r9c5 ==> r9c6 <> 4
interaction block b8 with column c5 for number 4 ==> r5c5 <> 4, r4c5 <> 4, r2c5 <> 4
nrczt-whip-cn[3]  n9{r8c1 r8c2} - n9{r6c2 r6c6} - {n9r5c5 .} ==> r2c1 <> 9
nrczt-whip-bn[4]  n3{r9c9 r8c9} - n8{r8c9 r8c2} - n9{r8c2 r8c1} - {n6r8c1 .} ==> r9c9 <> 6
nrczt-whip-rn[5]  n6{r7c7 r7c5} - {n6 n4}r8c5 - n4{r9c5 r9c9} - n3{r9c9 r9c6} - {n2r9c6 .} ==> r9c7 <> 6
nrczt-braid-rc[7]  n3{r9c9 r8c9} - {n3 n5}r8c6 - n5{r8c7 r3c7} - n7{r3c7 r1c7} - n5{r8c1 r5c1} - n7{r1c1 r9c1} - r9c3{n7 .} ==> r9c9 <>
nrczt-braid-rn[7]  n5{r5c1 r5c3} - n7{r5c3 r5c2} - n4{r5c1 r5c6} - n9{r6c1 r6c6} - {n4 n1}r3c6 - {n1 n9}r3c2 - n9r8{c2 .} ==> r5c1 <> 9
nrczt-braid-cn[8]  n8{r8c9 r7c9} - n8{r7c3 r5c3} - n5{r5c3 r5c1} - n4{r5c1 r5c6} - n4{r6c4 r1c4} - n3{r8c9 r9c9} - n4{r1c9 r3c9} - n5c9{
nrczt-braid-rc[8]  n5{r5c1 r5c3} - {n5 n7}r9c3 - n7{r9c1 r1c1} - n7{r3c3 r3c7} - n5{r3c7 r3c9} - n8{r5c3 r7c3} - n6{r5c7 r4c9} - r7c9{n8
nrczt-braid-rc[8]  n8{r8c9 r7c9} - n8{r7c3 r5c3} - n5{r5c3 r5c1} - n7{r5c1 r5c2} - n5{r3c9 r3c7} - n7{r3c2 r3c3} - {n7 n5}r7c3 - r9c3{n7
nrczt-braid-rc[8]  {n7 n5}r9c3 - n5{r9c1 r5c1} - n7{r5c1 r5c3} - n4{r5c3 r5c6} - n1{r9c2 r3c2} - {n1 n4}r2c1 - {n4 n9}r3c3 - r3c6{n9 .}
nrczt-whip-rc[9]  n5{r3c9 r3c7} - n7{r3c7 r1c7} - n9{r1c7 r5c7} - n1{r5c7 r2c7} - {n1 n4}r2c1 - {n4 n7}r3c3 - n7{r7c3 r9c1} - n1{r9c1 r9
nrczt-whip-rc[9]  n1{r9c2 r3c2} - n7{r3c2 r5c2} - n7{r5c1 r1c1} - n2{r1c1 r1c2} - n3{r1c2 r1c3} - n9{r1c3 r3c3} - {n9 n4}r4c3 - n4{r5c3
hidden-single-in-block b7 ==> r9c2 = 1
nrczt-whip-rn[6]  n5{r5c3 r5c1} - n7{r5c1 r5c2} - {n7 n9}r3c2 - n9{r3c3 r4c3} - n9{r6c2 r6c6} - {n8r6c6 .} ==> r5c3 <> 8
singles ==> r7c3 = 8, r8c9 = 8, r9c9 = 3, r8c6 = 3, r7c5 = 7
interaction row r7 with block b9 for number 6 ==> r8c7 <> 6
interaction column c9 with block b3 for number 4 ==> r3c7 <> 4, r2c7 <> 4, r1c7 <> 4
nrczt-whip-rn[2]  n4{r5c1 r5c6} - {n4r2c6 .} ==> r4c1 <> 4, r6c1 <> 4
nrct-chain[3]  {n9 n6}r6c1 - n6{r9c1 r8c2} - n9{r8c2 r8c1} ==> r4c1 <> 9, r1c1 <> 9
nrczt-whip-rc[3]  n6{r8c1 r8c2} - n9{r8c2 r8c1} - {n9r6c1 .} ==> r4c1 <> 6
singles ==> r4c1 = 2, r4c2 = 3, r1c2 = 2, r1c3 = 3
interaction block b1 with row r3 for number 9 ==> r3c7 <> 9, r3c6 <> 9
nrc-chain[3]  {n1 n4}r3c6 - n4{r2c6 r2c1} - n1{r2c1 r1c1} ==> r1c6 <> 1
nrc-chain[3]  n7{r1c7 r1c1} - n1{r1c1 r2c1} - {n1 n9}r2c7 ==> r1c7 <> 9
nrc-chain[2]  n9{r5c7 r2c7} - n9{r1c9 r1c6} ==> r5c6 <> 9
nrc-chain[3]  n9{r1c9 r1c6} - n6{r1c6 r1c4} - n6{r4c4 r4c9} ==> r4c9 <> 9
interaction block b6 with row r5 for number 9 ==> r5c5 <> 9, r5c3 <> 9, r5c2 <> 9
nrc-chain[3]  n1{r4c9 r4c5} - n9{r4c5 r6c6} - n9{r1c6 r1c9} ==> r1c9 <> 1
nrc-chain[2]  n7{r1c1 r1c7} - n1{r1c7 r1c1} ==> r1c1 <> 4
nrc-chain[3]  n9{r6c6 r4c5} - {n9 n4}r4c3 - n4{r4c8 r6c8} ==> r6c6 <> 4
nrc-chain[2]  n4{r6c4 r6c8} - n5{r6c8 r6c4} ==> r6c4 <> 6
nrc-chain[3]  n9{r6c6 r4c5} - n1{r4c5 r4c9} - n6{r4c9 r4c4} ==> r6c6 <> 6
interaction row r6 with block b4 for number 6 ==> r5c2 <> 6
xy-chain[3]  {n8 n1}r5c5 - {n1 n9}r4c5 - {n9 n8}r6c6 ==> r5c6 <> 8
xy-chain[3]  {n1 n8}r5c5 - {n8 n9}r6c6 - {n9 n1}r4c5 ==> r5c6 <> 1
interaction column c6 with block b2 for number 1 ==> r2c5 <> 1
xy-chain[3]  {n1 n9}r2c7 - {n9 n8}r2c5 - {n8 n1}r5c5 ==> r5c7 <> 1
interaction column c7 with block b3 for number 1 ==> r3c9 <> 1
nrc-chain[3]  n6{r1c6 r5c6} - {n6 n9}r5c7 - n9{r5c9 r1c9} ==> r1c6 <> 9
singles
GRID 4918 SOLVED. LEVEL = pB-NRCZT9, MOST COMPLEX RULE = NRCZT9
123456789
456789132
879231564
234697851
785314926
961528347
348972615
592163478
617845293
```

Back to top profile pm www

PIsaacson

Posted: Fri Nov 27, 2009 1:11 pm Post subject:

Denis,

Joined: 02 Jul 2008
Posts: 355
Location: Campbell, CA

It looks like your example above did not include locked sets, so I used the "-Xprn" to exclude them from my braids run:

Code:

```
000050000056700002800230060000007800000300020001020307340900010002100070000800090306431
```

```
12479     12379     3479      |46        5         14689     |1479      348       13489
149       5         6         |7         1489      1489      |149       348       2
8         179       479       |2         3         149       |14579     6         1459
--------- --------- ---------+--------- --------- ---------+--------- --------- ---------
24569     2369      3459      |456       1469      7         |8         45        14569
45679     6789      45789     |3         14689     145689    |14569     2         14569
4569      689       1         |456       2         45689     |3         45        7
--------- --------- ---------+--------- --------- ---------+--------- --------- ---------
3         4         578       |9         67        256       |256       1         568
569       689       2         |1         46        3456      |456       7         34568
1567      167       57        |8         467       23456     |2456      9         3456

 1) r2c8 <= 3 hidden single in r2
 2) r1c8 <= 8 hidden single in c8
 3) r5c6 <> 5 pointing pair b8/c6
 4) r6c6 <> 5 pointing pair b8/c6
 5) r4c9 <> 4 claiming pair b6/c8
 6) r5c7 <> 4 claiming pair b6/c8
 7) r5c9 <> 4 claiming pair b6/c8
 8) r4c9 <> 5 claiming pair b6/c8
 9) r5c7 <> 5 claiming pair b6/c8
10) r5c9 <> 5 claiming pair b6/c8
11) r4c1 <> 5 claiming pair b4/r5
12) r4c3 <> 5 claiming pair b4/r5
13) r6c1 <> 5 claiming pair b4/r5
14) r1c2 <> 1 nrc[2x2]-braid n2{r1c2 r4c2} - {n3r4c2 .}
15) r1c2 <> 7 nrc[2x2]-braid n2{r1c2 r4c2} - {n3r4c2 .}
16) r1c2 <> 9 nrc[2x2]-braid n2{r1c2 r4c2} - {n3r4c2 .}
17) r4c2 <> 6 nrc[2x2]-braid n2{r4c2 r1c2} - {n3r1c2 .}
18) r4c2 <> 9 nrc[2x2]-braid n2{r4c2 r1c2} - {n3r1c2 .}
19) r4c5 <> 6 nrc[2x2]-braid {n6 n7}r7c5 - {n7 n4}r9c5
20) r5c5 <> 6 nrc[2x2]-braid {n6 n7}r7c5 - {n7 n4}r9c5
21) r7c6 <> 6 nrc[1x1]-braid n6{r7c5 r4c5}
22) r8c6 <> 6 nrc[1x1]-braid n6{r7c5 r4c5}
23) r9c6 <> 4 nrc[2x2]-braid n2{r9c6 r7c6} - n5{r7c6 r8c6}
24) r9c6 <> 6 nrc[1x1]-braid n6{r7c5 r4c5}
25) r2c1 <> 9 nrc[3x3]-braid n9{r8c1 r8c2} - n9{r6c2 r6c6} - n9{r4c5 r5c5}
26) r2c5 <> 4 nrc[3x3]-braid {n4 n6}r8c5 - {n6 n7}r7c5 - {n7r9c5 .}
27) r4c5 <> 4 nrc[3x3]-braid {n4 n6}r8c5 - {n6 n7}r7c5 - {n7r9c5 .}
28) r5c5 <> 4 nrc[3x3]-braid {n4 n6}r8c5 - {n6 n7}r7c5 - {n7r9c5 .}
29) r8c6 <> 4 nrc[1x1]-braid n4{r8c5 r4c5}
30) r9c9 <> 6 nrc[3x3]-braid n6{r7c7 r7c5} - n7{r7c5 r9c5} - n4{r9c5 r9c7}
31) r5c1 <> 9 nrc[4x5]-braid n5{r5c1 r5c3} - n4{r5c3 r5c6} - {n4 n1}r3c6 - {n1r3c2 .}
                 [1]-strand {. n9r5c1} - n9{r6c1 r6c6}
32) r9c7 <> 6 nrc[3x5]-braid n6{r7c7 r7c5} - n7{r7c5 r9c5} - n4{r9c5 r9c7}
                 [2]-strand {. n6r9c7} - n2{r9c7 r9c6} - n3{r9c6 r9c9}
33) r9c9 <> 5 nrc[4x5]-braid n5{r7c7 r3c7} - n7{r3c7 r1c7} - n7{r1c1 r5c1} - {n5r5c1 .}
                 [1]-strand {. n5r9c9} - {n5 n7}r9c3
34) r5c1 <> 6 nrc[4x7]-braid n5{r5c1 r5c3} - n7{r5c3 r5c2} - n7{r3c2 r3c7} - n5{r3c7 r7c7}
                 [3]-strand {. n6r5c1} - n5{r5c1 r5c3} - n8{r5c3 r7c3} - {n8 n5}r7c9
                 [2]-strand {. n6r5c1} - n5{r5c1 r5c3} - {n5 n7}r9c3
35) r8c1 <> 6 nrc[5x7]-braid n9{r8c1 r8c2} - n8{r8c2 r8c9} - {n8 n6}r7c9 - n6{r4c9 r4c4} - {n6 n4}r1c4
                 [2]-strand {. n6r8c1} - {n6 n4}r8c5 - {n4 n5}r8c7
36) r8c9 <> 6 nrc[4x7]-braid {n6 n4}r8c5 - n4{r8c7 r9c7} - n4{r1c7 r1c9} - {n4 n6}r1c4
                 [2]-strand {. n6r8c9} - n8{r8c9 r7c9} - n5{r7c9 r3c9}
                 [1]-strand {. n6r8c9} - n3{r8c9 r9c9}
37) r2c5 <> 1 nrc[4x5]-braid n1{r4c5 r5c6} - n4{r5c6 r5c3} - n5{r5c3 r5c1} - {n5 n9}r8c1
                 [1]-strand {. n1r2c5} - {n1 n4}r2c1
38) r5c6 <> 1 nrc[1x1]-braid n1{r4c5 r5c5}
39) r5c1 <> 4 nrc[6x8]-braid n5{r5c1 r5c3} - n7{r5c3 r5c2} - {n7 n9}r3c2 - n9{r1c3 r4c3} - {n9 n1}r4c5 - {n1 n6}r4c9
                 [2]-strand {. n4r5c1} - n5{r5c1 r5c3} - {n5 n7}r9c3
                 [1]-strand {. n4r5c1} - {n4 n1}r2c1
40) r5c3 <> 4 nrc[6x8]-braid n5{r5c3 r5c1} - n7{r5c1 r5c2} - n8{r5c2 r6c2} - n9{r6c2 r4c3} - {n9 n1}r4c5 - {n1 n6}r4c9
                 [2]-strand {. n4r5c3} - n5{r5c3 r5c1} - {n5 n9}r8c1
                 [1]-strand {. n4r5c3} - n8{r5c3 r7c3}
41) r5c6 <= 4 hidden single in r5
42) r1c4 <= 4 hidden single in b2
43) r1c6 <= 6 hidden single in b2
44) r1c1 <> 9 nrc[4x8]-braid {n9 n5}r8c1 - {n5 n3}r8c6 - n3{r8c9 r9c9} - n4{r9c9 r8c9}
                 [3]-strand {. n9r1c1} - {n9 n5}r8c1 - {n5 n7}r9c3 - {n7 n4}r3c3
                 [2]-strand {. n9r1c1} - n9{r8c1 r8c2} - n8{r8c2 r8c9}
45) r2c6 <> 1 nrc[3x3]-braid {n1 n4}r2c1 - n4{r6c1 r4c3} - n3{r4c3 r1c3}
46) r3c6 <= 1 hidden single in b2
47) r9c2 <= 1 hidden single in c2
48) r2c7 <> 9 pointing pair b2/r2
49) r1c1 <> 7 nrc[4x6]-braid n1{r1c1 r2c1} - n4{r2c1 r6c1} - n6{r6c1 r6c2} - n8{r6c2 r6c6}
                 [1]-strand {. n7r1c1} - n2{r1c1 r4c1}
                 [1]-strand {. n7r1c1} - n7{r3c2 r5c2}
50) r1c3 <> 7 nrc[5x7]-braid n3{r1c3 r4c3} - n4{r4c3 r3c3} - {n4 n1}r2c1 - n1{r2c7 r1c9} - {n1r5c9 .}
                 [3]-strand {. n7r1c3} - n3{r1c3 r4c3} - n9{r4c3 r5c3} - n9{r5c7 r1c7}
51) r1c7 <= 7 hidden single in r1
52) r3c2 <> 9 nrc[3x4]-braid n9{r8c2 r8c1} - n9{r6c1 r4c3} - {n4r4c3 .}
                 [1]-strand {. n9r3c2} - n9{r3c7 r5c7}
53) r3c2 <= 7 naked single
54) r4c3 <> 9 pointing pair b1/c3
55) r5c3 <> 9 pointing pair b1/c3
56) r3c7 <> 4 nrc[6x8]-braid n4{r3c3 r4c3} - {n4 n5}r4c8 - {n5 n6}r4c4 - n6{r4c9 r5c9} - {n6 n8}r7c9 - n8{r8c9 r8c2}
                 [1]-strand {. n4r3c7} - n5{r3c7 r3c9}
                 [1]-strand {. n4r3c7} - n9{r3c7 r5c7}
57) r3c9 <> 9 nrc[2x3]-braid n9{r4c9 r5c7} - n1{r5c7 r4c9}
                 [1]-strand {. n9r3c9} - {n9 n1}r1c9
58) r4c1 <> 6 nrc[4x7]-braid {n6 n5}r4c4 - {n5 n4}r4c8 - n4{r4c3 r3c3} - n4{r3c9 r8c9}
                 [3]-strand {. n6r4c1} - n6{r5c2 r8c2} - n8{r8c2 r8c9} - n3{r8c9 r9c9}
59) r4c4 <> 5 nrc[5x6]-braid n6{r4c4 r4c9} - n1{r4c9 r4c5} - n9{r4c5 r4c1} - n9{r8c1 r8c2} - n8{r8c2 r8c9}
                 [2]-strand {. n5r4c4} - n6{r4c4 r4c9} - n6{r5c7 r5c2}
60) r4c4 <= 6 naked single
61) r6c4 <= 5 naked single
62) r6c8 <= 4 naked single
63) r4c8 <= 5 hidden single in r4
64) r5c2 <> 6 pointing pair b6/r5
65) r1c9 <> 1 nrc[6x8]-braid {n1 n9}r4c9 - {n9 n6}r5c9 - {n6 n8}r7c9 - n8{r8c9 r8c2} - {n8 n9}r5c2 - {n9r5c5 .}
                 [2]-strand {. n1r1c9} - n9{r1c9 r3c7} - n5{r3c7 r3c9}

singles from here on

123456789456789132879231564234697851785314926961528347348972615592163478617845293 puzzle 1 givens 26 nrczt 8.0       302.5493 m
```

I'm still working on the display and I know it's ugly, but it should be much easier to read than my prior versions. Some pointers:
a) The initial braid on the step line " nnn) rrcc <> d nrc[NxM]-braid" uses the number N to indicate the length of the primary strand that detected the conflict. The number M indic
support the braid. If N=M, then it's really just a standard whip and there should be no supporting strands listed after it.
b) Supporting strands follow the initial braid and they are presented in descending length. For reasons hard to explain, each begins with the z-target, which you can ignore. You ha
original braid in order to "see" the common runs and unique truths.

The final score for my solution is 8. I ran the braids through a separate validation program and it concurs that they are all individually valid, so I'm slowly comparing each one ma

Cheers,
Paul

Back to top profile pm

denis_berthier

Joined: 19 Jun 2007
Posts: 1123
Location: Paris, France

Posted: Mon Nov 30, 2009 8:37 am Post subject:

Allan,

For any puzzle P, one can define the minimal depth of T&E (in my sense, as defined at the start of this thread) necessary to solve it. **T&E-depth(P) is an intrinsic property of**
I've shown that all the puzzles produced by random generators, in a total sample size of ~ 10 million, have depth 0 or 1.
Based on gsf's collection of hardest, I've also shown that only 5537 known puzzles have depth 2 and no known puzzle has depth > 2 (even the few with backdoor-size 3).
Of course, this doesn't prove that there can't exist puzzles at depth 3, but if there are, they must be very rare.

Now, considering your approach, you have a notion of rank and, for each puzzle P, one can define **rank(P), an intrinsic property of P**: it is the smallest r such that P can be sol
rank ≤ r.
AFAIK, all the patterns you've been using have ranks 0, 1 or 2.

I'm therefore wondering (just a question, not yet a conjecture): do we have **T&E-depth(P) = rank(P)?**

How could this be tested? It depends on your answer to the following question: can you disable rank 2 (and above) patterns in your solver?

If so, there are several tests we could do:
- you could check whether you can solve all the puzzles in Sudogen0_1M at ranks 0 or 1 - or even all the puzzles published on my web pages for all the kinds of generators. (Sorry
checked this by solving all of them with nrczt-whips.)[Edit: on second thoughts,this is useless, as it should be easy to prove that any braid has rank 1.]
- we could try gsf's collection and compare the lists of puzzles solved at depths 0/1 and with ranks 0/1.

Back to top profile pm www

Allan Barker

Joined: 21 Feb 2008
Posts: 485
Location: Bangkok

Posted: Mon Nov 30, 2009 5:06 pm Post subject:

Hi Denis,

> **Denis Berthier wrote:**
> Now, considering your approach, you have a notion of rank and, for each puzzle P, one can define rank(P), an intrinsic property of P: it is the smallest r such that P ca
> (in your sense) of rank ≤ r.
> AFAIK, all the patterns you've been using have ranks 0, 1 or 2.
>
> I'm therefore wondering (just a question, not yet a conjecture): do we have **T&E-depth(P) = rank(P)?**

I don't see a connection between rank, which is a static property of logic, and the number of assumed clues needed for a TE backtrack algorithm to solve a puzzle. For example, a
backtracking(n=0) doesn't compute ☹.

But rank does represent a degree of freedom of sorts and rank 2 can eliminate some candidates that rank 1 can't, so there must be a relation between rank and solvability, as you

> **Denis Berthier wrote:**
> How could this be tested?

As far as a relation to T&E, **T&E-depth(0) <> rank(0)** , so no comparison. Rank and solvability is a different question.

> **Denis Berthier wrote:**
> can you disable rank 2 (and above) patterns in your solver?

Yes, but other properties may overshadow the effects of rank in any kind of large batch testing. Group links are a good example, as they can produce shorter paths with little logic
different methods/rank on a puzzle by puzzle or even a candidate by candidate basis. This way you see what logical form makes a difference. I think I can find a couple of good ex

Back to top profile pm www

denis_berthier

Joined: 19 Jun 2007
Posts: 1123
Location: Paris, France

Posted: Mon Nov 30, 2009 6:35 pm Post subject:

Allan,
OK, the fish example is convincing and the equality can't hold.

Then take it in either of these ways:
- what's the smallest SER of puzzles that can't be solved with only patterns of rank 0 or 1?
- how many puzzles do you know that can't be solved with only patterns of rank 0 or 1?

Back to top profile pm www

Allan Barker

Joined: 21 Feb 2008
Posts: 485
Location: Bangkok

Posted: Fri Dec 04, 2009 4:24 am Post subject:

> **denis_berthier wrote:**
> **Allan**,
> OK, the fish example is convincing and the equality can't hold.
>
> Then take it in either of these ways:
>
> - what's the smallest SER of puzzles that can't be solved with only patterns of rank 0 or 1?
> - how many puzzles do you know that can't be solved with only patterns of rank 0 or 1?

Sorry for the delay, I wanted to finish up some things first, see http://www.sudoku.com/boards/viewtopic.php?p=84253#84253
for an answer.

Back to top profile pm www

denis_berthier

Posted: Fri Dec 04, 2009 7:17 am Post subject:

I can't see there any answer to my question.

Joined: 19 Jun 2007
Posts: 1123
Location: Paris, France

The kind of answer I was expecting is something like the table here:
http://www.sudoku.com/boards/viewtopic.php?t=6390&postdays=0&postorder=asc&start=44.
It shows that almost all (and probably all) known puzzles can be solved at rank 0 or 1.

Back to top

profile pm www

Allan Barker

Posted: Sat Dec 12, 2009 2:43 pm Post subject:

Joined: 21 Feb 2008
Posts: 485
Location: Bangkok

denis_berthier wrote:

> Remember that gsf's collection is ordered according to decreasing Q1 rating.
> In the following table, the first column defines the sets of puzzles I consider: [....................................]
>
> **Code:**
>
> ```
> Nb of puzzles solved:
>
> FP family............ECP+NS+HS.........ECP+NS+HS+BI...............SubsetRules+BI
> Braids used..........nrczt-braids......hinged-zt-braids...........hinged-zt-LS-braids
> 1-500................0.................187........................443
> 501-1000.............0.................178........................460
> 1001-2000............0.................331........................976
> 2001-3000............0.................251........................953
> 3001-4000............1.................233........................945
> 4001-5000............1.................333........................930
> 5001-5500............47................348........................500
> 5501-6000............434...............490........................500
> 6001-7000............981...............1000.......................1000
> 7001-8152............1152..............1152.......................1152
> Total................2616..............4505.......................7859
> Rest.................5536..............3647.......................393
> ```
>
> When the theory T in T&E(T) becomes more complex than ECP+NS+HS+BI, computation times are very long: Triplets and Quads are relatively rare patterns and, in or
> procedure must explore almost all the candidate hypotheses, sometimes several times (phase iteration). After puzzles solved by hinged-zt-braids are discarded, there
> collection; for each of them, T&E must try between 500 and 1,500 auxiliary puzzles; this makes ~ 3 to 4 million puzzles to deal with using the rules in T.

Denis,

As this seems to be the current subject, I have been trying to better understand some of the details. Most is clear but at the point referenced above, it's not clear to me which of t
puzzles.

Were they solved,

1. using a not T&E solver that implements the full FP family or
2. using the T&E(FP) procedure discussed above, and applying the T&E theorem, or
3. using 1 or 2 depending on computational necessity?

This is purely computational question, I understand the T&E theorem and how it's applied, and also know this is not the final data.

Allan

Back to top

profile pm www

denis_berthier

Posted: Sat Dec 12, 2009 3:13 pm Post subject:

Joined: 19 Jun 2007
Posts: 1123
Location: Paris, France

Allan Barker wrote:

> As this seems to be the current subject, I have been trying to better understand some of the details. Most is clear but at the point referenced above, it's not clear to m
> was use to solve the puzzles.
> Were they solved,
> 1. using a not T&E solver that implements the full FP family or
> 2. using the T&E(FP) procedure discussed above, and applying the T&E theorem, or
> 3. using 1 or 2 depending on computational necessity?
> This is purely computational question, I understand the T&E theorem and how it's applied, and also know this is not the final data.

As has been clearly stated, all these data were obtained using the T&E(FP) vs braids(FP) equivalence theorem and computed with the T&E(FP) procedure.
As I have also clearly stated, neither the zt-braids(FP) nor the zt-whips(FP) have been implemented in SudoRules, except of course for the basic FP={NS HS}. They have been pa

So, if you imply that what you're trying to do in the "ribbon" thread is an implementation of zt-whips(FP), which are precisely defined patterns but have never been fully implemen

Last edited by denis_berthier on Sat Dec 12, 2009 5:10 pm; edited 1 time in total

Back to top

profile pm www

denis_berthier

Posted: Sat Dec 12, 2009 4:54 pm Post subject:

[deleted: edit error]

Joined: 19 Jun 2007
Posts: 1123
Location: Paris, France

Last edited by denis_berthier on Sat Dec 12, 2009 5:11 pm; edited 1 time in total

Back to top

profile pm www

Allan Barker

Posted: Sat Dec 12, 2009 5:08 pm Post subject:

Joined: 21 Feb 2008
Posts: 485
Location: Bangkok

denis_berthier wrote:

> **Allan Barker wrote:**
>
> > As this seems to be the current subject, I have been trying to better understand some of the details. Most is clear but at the point referenced above, it's no
> > the two approaches was use to solve the puzzles.
> > Were they solved,
> > 1. using a not T&E solver that implements the full FP family or
> > 2. using the T&E(FP) procedure discussed above, and applying the T&E theorem, or
> > 3. using 1 or 2 depending on computational necessity?

> > This is purely computational question, I understand the T&E theorem and how it's applied, and also know this is not the final data.
>
> As has been clearly stated, all these data were obtained using the T&E(FP) vs braids(FP) equivalence theorem and computed with the T&E(FP) procedure.
> As I have also clearly stated, neither the zt-braids(FP) nor the zt-whips(FP) have been implemented in SudoRules. They have been partly implemented in Paul's solver

OK, that's clear. If you read the written portion of what I quoted, the wording leads me to think it was a mixture, possibly with Paul's help.

> **denis_berthier wrote:**
>
> So, if you imply that what you're trying to do in the "ribbon" thread is an implementation of zt-whips(FP), which are precisely defined patterns but have never been fu
> agree.

I have no need to make implications and prefer to speak directly. The reason I am trying to learn more is to see how we would be able to make any comparisons, if we want to.

Cheers.

Back to top profile pm www

denis_berthier

Joined: 19 Jun 2007
Posts: 1123
Location: Paris, France

Posted: Sat Dec 12, 2009 5:26 pm Post subject:

Paul wasn't informed of my work on whips(FP) and braids(FP) (definitions, theorems, stats) before I published it.

> **Allan Barker wrote:**
>
> The reason I am trying to learn more is to see how we would be able to make any comparisons, if we want to.

I have given the basis for such comparisons in your "ribbon" thread.
It is obvious and it is based on the lexicon already mentioned there (which you haven't denied):

look-back => using the (z)t extension
Truth => the content of a 2D-cell (i.e. rc-, rn-, cn- or bn- cell)
overlapping truths => different 2D-cells sharing (at least) a candidate (cells which must therefore be in different 2D spaces)
logic => pattern
logic integration => accepting patterns as right-linking objects instead of mere right-linking candidates - as in zt-whips(FP).

They show clearly that, when your renaming of everything is cancelled and except perhaps a few cases of doubly linked AAAAL/HS, your "ribbons", whose fundamental property is
not different from zt-whips(FP) or zt-braids(FP) if you allow branching.

Back to top profile pm www

Mauricio

Joined: 22 Mar 2006
Posts: 1076

Posted: Fri Jan 08, 2010 7:19 pm Post subject:

Here is a procedure that calculates a lower bound of the pBrating of a puzzle.

I will use some of Denis notations (nrc linked, 2D cell).

For a candidate R0^1, an **MBraid** (M for Mauricio) of length **n** for R0^1 is the following sequence of sets of candidates {R0^1},{R1^1,R1^2,...,R1^m1},{R2^1,...,R2^m2},...,{R
properties:

1. For every i>0, {Ri^1,Ri^2,...,Ri^mi} are ***all*** the possible candidates such that for every Ri^j, there exists a 2D cell that contains it and that every other candidate in this ce
k<i (k=0 is possible).
2. Ri^j is different from a candidate Rk^l if i<>k and j<>l.
3. For every pair 0<=k<=i<**n**, a candidate Ri^j is never nrc linked to a candidate Rk^l.
4. There exists a candidate R**n**^j that is nrc linked to a candidate Rk^l (k=**n** and k =0 are possible).

If there exists a MBraid for R0^1 and the candidate R0^1 were true, then all candidates Ri^j would be true and we would reach a contradiction, so R0^1 can be eliminated.

What I do with that procedure is assert R0^1, then make all ECP eliminations, then some singles are discovered, we tag them all R1^1,R1^2,...R1^m1, and we assert them all at
new singles are discovered, we tag them R2^1,R2^2,...,R2^m2 and assert them all at once, and so on until a contradiction is reached.

If R0^1 can be eliminated using T&E(Ns,Hs), then there exists a unique MBraid that eliminates it, and we know also that there exists a braid that eliminates it, and moreover, the
not greater than the length of any braid that eliminates it.

The candidates R1^1,R1^2,...,R1^m1,R2^1,...,R2m2,R3^1,...,R**n**^m**n** are the right linking candidates of a braid that eliminates R0^1.

The MBraid rating of a puzzle if the minimum **n** such that the puzzle can be solved using MBraid of length at most **n**, if no **n** satisfies the condition, then we say that the rating is i
using MBraids). The MBraid rating of a puzzle is finite if and only if it is solvable using T&E(NS,HS), and the last part is true if and only if it is solvable using braids.

The following puzzle has a MBraid rating of 20 (19?)

Code:

```
000001002000030040005600700002800003040050060900007400007003004030020080600400300
```

and so it can be solved using braids but it can't be solved using braids of length 19 or less (BTW, it is not so hard if you use some exotic uniqueness argument 😁).

Notes:

1. I developed this before Denis published his definition of braids, only a few days ago I realized it gives a lower bound of the pB rating of a puzzle.
2. Finding minimal braids is not an easy task, and MBraids contain a lot of unnecessary information, and moreover, a Braid can be complexly embedded in a MBraid. Generally
than the MBraid rating.
3. MBraid ratings can be calculated very quickly
4. I don't think this procedure can be easily turned into a pBraids rating calculator.

Back to top profile pm

Red Ed

Posted: Fri Jan 08, 2010 7:39 pm Post subject:

Joined: 06 Jun 2005
Posts: 1007

> **Mauricio wrote:**
>
> The following puzzle has a MBraid rating of 20 (19?)
>
> **Code:**
>
> ```
> 000001002000030040005600700002800003040050060900007400007003004030020080600400300
> ```
>
> and so it can be solved using braids but it can't be solved using braids of length 19 or less (BTW, it is not so hard if you use some exotic uniqueness argument 😀)

What "it" is "not so hard"? I like exotic uniqueness arguments and don't want to feel I'm missing out ...!

Back to top profile pm

Mauricio

Posted: Fri Jan 08, 2010 9:38 pm Post subject:

Joined: 22 Mar 2006
Posts: 1076

> **Red Ed wrote:**
>
> > **Mauricio wrote:**
> >
> > The following puzzle has a MBraid rating of 20 (19?)
> >
> > **Code:**
> >
> > ```
> > 000001002000030040005600700002800003040050060900007400007003004030020080600400300
> > ```
> >
> > and so it can be solved using braids but it can't be solved using braids of length 19 or less (BTW, it is not so hard if you use some exotic uniqueness argume
>
> What "it" is "not so hard"? I like exotic uniqueness arguments and don't want to feel I'm missing out ...!

The puzzle is automorphic (reflex along main diagonal and swap digits 1-9,2-6,3-4), then you can make some eliminations in the diagonal (explicitely, you can eliminate 1,2,3,4,6, solved using singles only).

Back to top profile pm

Display posts from previous: All Posts Oldest First Go

new topic post reply **Sudoku Players' Forums Forum Index -> Advanced solving techniques**

Goto page

Page 9 of 10

Stop watching this topic

Jump to: Advanced sc