Nick Morris

Exam 2 Take Home – Operations Research

Professor Proaño

5/22/14

Sudoku Model

```
set N;                          #set of Numbers
set R;                          #set of Rows
set C;                          #set of Columns

param v{ N };                   #a Value 1..N
param g{ R,C };                 #Given Sudoku
param s;                        #Size of an (s x s) Sudoku

var x{ N, R, C } binary;        #a Number in N that does or doesn't go in a Row-Column cross section
var A{ r in R, c in C };        #Solved Sudoku

maximize Sudoku: sum{ n in N, r in R, c in C }( v[ n ]*x[ n, c, r ] );
s.t. Grids{ r in R, c in C }: sum{ n in N }x[ n, r, c ] = 1;
s.t. Rows{ n in N, r in R }: sum{ c in C }x[ n, r, c ] = 1;
s.t. Columns{ n in N, c in C }: sum{ r in R }x[ n, r, c ] = 1;
s.t. Boxes{ n in N, m in 0..(sqrt(s) - 1), q in 0..(sqrt(s) - 1) }:
sum{ r in (sqrt(s)*m + 1) .. (sqrt(s)*m + sqrt(s)), c in (sqrt(s)*q + 1) .. (sqrt(s)*q + sqrt(s)) }x[ n, r ,c ] = 1;
s.t. AValue{ r in R, c in C }: A[ r, c ] = sum{ n in N }( v[ n ]*x[ n, r, c ] );
```

Sudoku Data

set N := 1 2 3 4 5 6 7 8 9;

set R := 1 2 3 4 5 6 7 8 9;

set C := 1 2 3 4 5 6 7 8 9;

param v :=

1 1

2 2

3 3

4 4

5 5

6 6

7 7

8 8

9 9;

param g:	1	2	3	4	5	6	7	8	9 :=
1	6	0	0	3	0	0	1	0	0
2	0	7	1	6	2	0	0	0	0
3	8	0	5	0	0	1	0	0	0
4	5	0	0	8	7	0	9	0	1
5	0	0	9	0	0	0	6	0	0
6	4	0	7	0	6	9	0	0	8
7	0	0	0	2	0	0	8	0	7
8	0	0	0	0	8	6	4	1	0
9	0	0	8	0	0	3	0	0	2;

param s := 9;

Sudoku Run

```
reset;

model Sudoku.mod;

data Sudoku.dat;

option solver cplex;

option cplex_options 'sensitivity';

for{ r in R, c in C }

{

        if g[ r, c ] != 0 then

                {

                        fix A[ r, c ] := g[ r, c ];

                }

}

solve;

display A;
```

Solution

```
ampl: include Sudoku.run;
CPLEX 12.6.0.0: sensitivity
CPLEX 12.6.0.0: optimal integer solution; objective 405
0 MIP simplex iterations
0 branch-and-bound nodes

suffix up OUT;
suffix down OUT;
suffix current OUT;
A [*,*]
:   1   2   3   4   5   6   7   8   9    :=
1   6   2   4   3   5   7   1   8   9
2   9   7   1   6   2   8   3   5   4
3   8   3   5   4   9   1   7   2   6
4   5   6   3   8   7   2   9   4   1
5   2   8   9   1   3   4   6   7   5
6   4   1   7   5   6   9   2   3   8
7   3   4   6   2   1   5   8   9   7
8   7   5   2   9   8   6   4   1   3
9   1   9   8   7   4   3   5   6   2
;

ampl:
```