LINEAR PROGRAMMING SUDOKU SOLVER

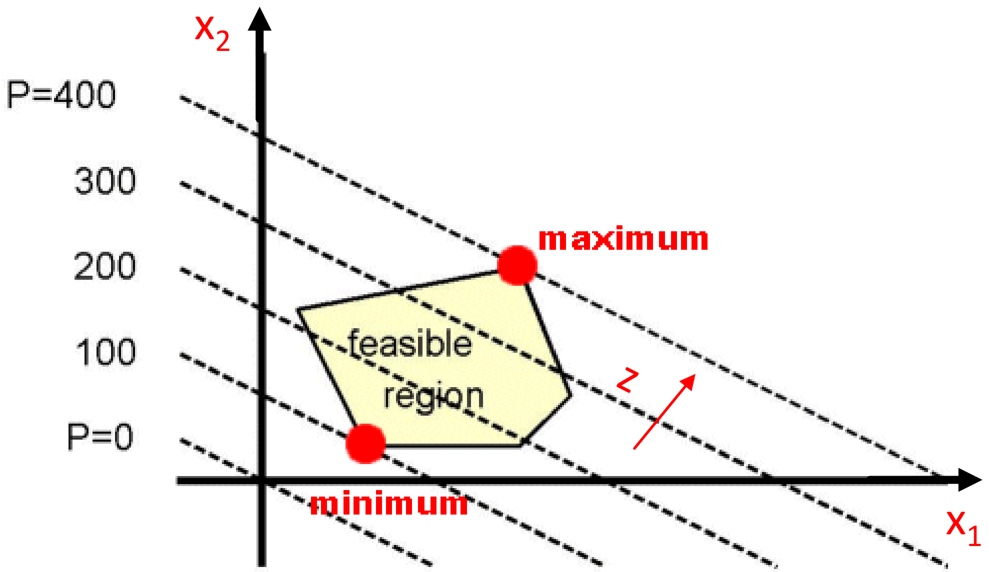
Anna Petrone
DC Hack && Tell
6/7/2016

MY RELATIONSHIP WITH SUDOKU …

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

LINEAR PROGRAMMING

Feasible region



Formulation

Max/min $\quad z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$

subject to:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \ (\le, =, \ge)\ b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \ (\le, =, \ge)\ b_2 \\ \vdots \\ a_{m1}x1 + a_{m2}x_2 + \ldots + a_{mn}x_n \ (\le, =, \ge)\ b_m \end{cases}$$

x_j = decision variables
b_i = constraint levels
c_j = objective function coefficients
a_{ij} = constraint coefficients

SUDOKU LP DECISION VARIABLES

Binary decision variables

$$x_{ij}^{1} = \begin{cases} 1 & \text{if row } i\text{, column } j \text{ is a } 1 \\ 0 & \text{if not} \end{cases}$$

$$x_{ij}^{2} = \begin{cases} 1 & \text{if row } i\text{, column } j \text{ is a } 2 \\ 0 & \text{if not} \end{cases}$$

$\vdots$

Notation

j = 1,2,3,…N

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

i = 1,2,3,…N

LP CONSTRAINTS

Let $\mathcal{N} = \{1, 2, ...N\}$

satisfy game rules:

- Each row must have the numbers 1, 2, … N **exactly once**

$$\forall \quad p \in \mathcal{N}:$$
$$\forall \quad i \in \mathcal{N}:$$
$$\sum_{j \in \mathcal{N}} x_{ij}^p = 1$$

- Each column must have the numbers 1, 2, … N **exactly once**
  - [similar]
- Each box must have the numbers 1, 2, … N **exactly once**
  - [tricky]

"gotcha" constraint

- **Each cell must be assigned a single value**

$$\forall \quad i \in \mathcal{N}:$$
$$\forall \quad j \in \mathcal{N}:$$
$$\sum_{p \in \mathcal{N}} x_{ij}^p = 1$$

CPLEX IMPLEMENTATION

Note there is no objective function – *any* feasible solution will do the trick.

This is called a <u>constraint programming</u> problem

sudoku.mod

```
int ndigits = 9; // update the box dimension
int sqrt_ndigits = 3; // this number should be the square root of the above number

range N = 1..ndigits;
range Ns = 1..sqrt_ndigits;

dvar int+ x[N][N][N] in 0..1;

minimize 0;
subject to{

   forall(val in N){
       forall(row in N){
           sum(i in N)(x[val][row][i]) == 1;
        }

       forall(col in N){
           sum(i in N)(x[val][i][col]) == 1 ;
       }

       forall(b1 in Ns){
           forall(b2 in Ns){
               sum(i in Ns)(sum(j in Ns)(x[val][i + (b1-1)*sqrt_ndigits ][j+ (b2-1)*sqrt_ndigits ])
           }
       }

   }

   forall(row in N){
       forall(col in N){
           sum(val in N)(x[val][row][col]) == 1; // cant assign more than one value to a cell
       }

   }
// here add the values that were initially specified.
//x[1][2][3]==1; // means row 2, column 3 is a 1
}
```

RESULTS

- Finds solution for 9x9, 16x16, and 25x25 grids in **a few seconds**
- 64 x 64 couldn't solve after 1 hour!

N	rows (N^4 + N^2)	columns (N^3)	non zeros coeffs	pre-solve time (ticks)	total time (ticks)
4	64	64	256	0.12	0.36
9	324	729	2,916	1.31	3.64
16	1,024	4096	16,384	7.06	20.41
25	2,500	15625	62,500	26.65	2137.18
64	16,384	262,144	1,048,576	451.51	
81					

FUTURE QUESTIONS

- What's the minimum number of cells you must pre-specify in order for the solution to be unique?
- More generally, what is the relationship between the number of cells initially specified and the number of possible solutions?
- Quantifying difficulty level
  - Follow up: can you judge difficulty just from knowing initial number of specified cells?
- Reformulation as column generation problem for solving for large N
  - Incrementally adds variables according to a "pricing" sub-problem

THANKS!

Anna Petrone
Github.com/AMPETR/sudoku
Twitter @1littlevictory