

**YRKESHÖGSKOLAN ARCADA**

# GRUPPARBETE OPTIMERING

10.12.2021

Fredrik Ståhl, Johan Penttinen, Kristoffer Kuvaja Adolfsson  
Sudoku med olika modeller och en jämförelse av resutlaten



# Problemformulering

## Optimera vad?

- Python
- LP
  - Lpsolve
  - PuLP
- Backtracking
- GA





# Sudoku definiering

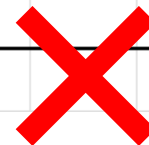
# Sudoku

- 9 x 9 rutnät på 81 rutor
- en låda består av 3 x 3 rutnät
- latinsk kvadrat
- fyra kategorier

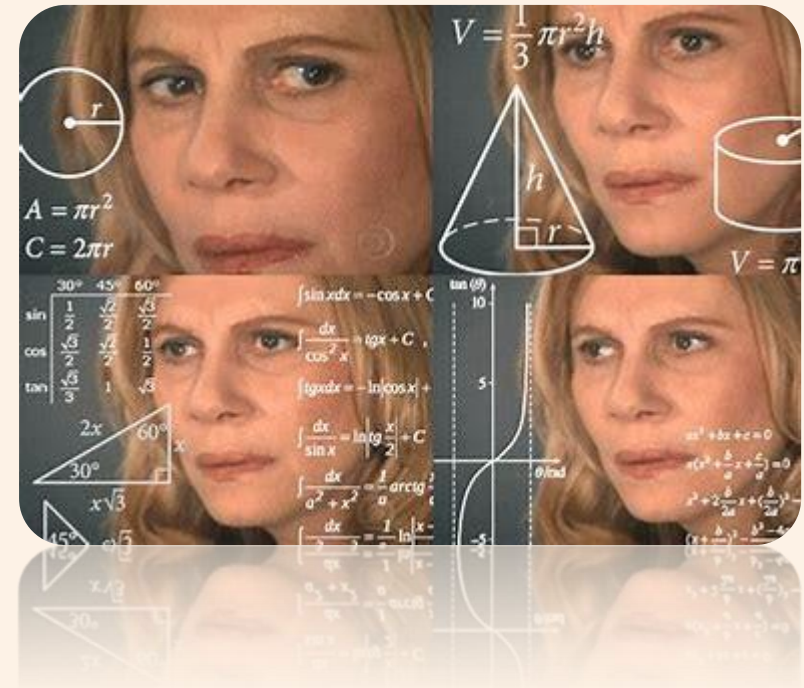
	8	3		2	1			7
9	6		3		5	8	2	1
2	5	1			6		9	3
	4	8	1		2		7	
	2	9			4		3	
	3		7		8		4	
3	7				9		1	4
8	1		2	5	3		6	9
6	9		4	1	7		8	2

# Lämpliga sudokun

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



# Lösningar & Optimeringar





# Ipsolve



## Variabler och Begränsningar

- Binära
- Celler
- Lådor
- Kolumner
- Rader
- Totalt:
- 729 binära variabler
- 324 begränsningar

- 1 cell – vi ändrar endast värdet
- $x_{111} + x_{211} + x_{311} + x_{411} + x_{511} + x_{611} + x_{711} + x_{811} + x_{911} = 1$
- 1 rad – vi ändrar på kolumnen
- $x_{111} + x_{112} + x_{113} + x_{114} + x_{115} + x_{116} + x_{117} + x_{118} + x_{119} = 1$
- 1 kolumn – vi ändrar på raden
- $x_{111} + x_{121} + x_{131} + x_{141} + x_{151} + x_{161} + x_{171} + x_{181} + x_{191} = 1$
- 1 låda – 3 x 3
- $x_{111} + x_{112} + x_{113} + x_{121} + x_{122} + x_{123} + x_{131} + x_{132} + x_{133} = 1$

## Sudoku begränsningar

- x812
- x313
- Osv..
- x299

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

# Ipsolve filen

- Bara skriva 1053 rader
- Python!

# Lpsolve – Easy 02

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

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

## Lpsolve - resultat

- Resultat
- Easy 02
  - 0.031 seconds
- Expert 04
  - $\infty$

```
Log Messages
Model name: 'LPSolver' - run #1
Objective: Minimize (R0)

SUBMITTED
Model size:      324 constraints,      729 variables,      2916 non-zeros.
Sets:           0 GUB,                0 SOS.

Using DUAL simplex for phase 1 and PRIMAL simplex for phase 2.
The primal and dual simplex pricing strategy set to 'Devex'.
```

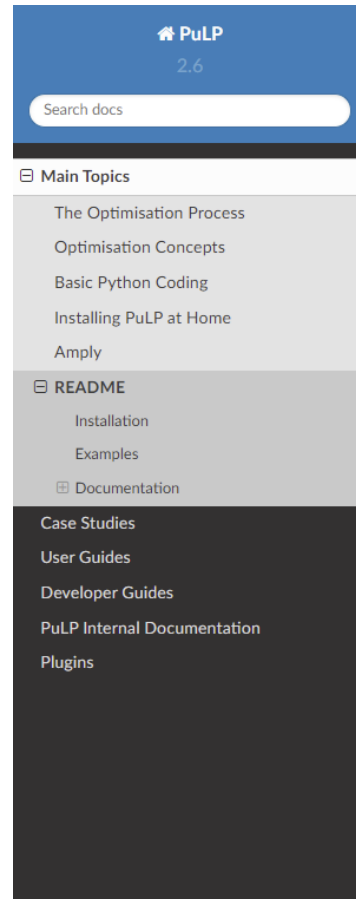
1:1	ITE: 796626451	IPS: 26405	INV: 12682070	NOD: 0	TME: 30168.82
-----	----------------	------------	---------------	--------	---------------



# PuLP – The Python solver!

# PuLP

- Linjär
- Syntax
- Modeller
- Logik och begränsningar



» Main Topics » pulp

[View page source](#)

## pulp

build passing

PuLP is an LP modeler written in Python. PuLP can generate MPS or LP files and call [GLPK](#), COIN-OR CLP/CBC, [CPLEX](#), [GUROBI](#), [MOSEK](#), [XPRESS](#), [CHOCO](#), [MIPCL](#), [SCIP](#) to solve linear problems.

### Installation

The easiest way to install pulp is via [PyPi](#)

If pip is available on your system:

```
python -m pip install pulp
```

Otherwise follow the download instructions on the PyPi page.

If you want to install the latest version from github you can run the following:

```
python -m pip install -U git+https://github.com/coin-or/pulp
```

On Linux and OSX systems the tests must be run to make the default solver executable.

```
sudo pulptest
```



# PuLP – Expert 04

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

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

## PuLP - resultat

- GLPSOL--GLPK LP/MIP Solver 5.0
- Parameter(s) specified in the command line:
- --cpxlp
- Scaling...
- A:  $\min|a_{ij}| = 1.000e+00$   $\max|a_{ij}| = 1.000e+00$  ratio = 1.000e+00
- Size of triangular part is 136
- Solving LP relaxation...

```
[ [5 2 1 3 8 9 7 4 6]
  [4 3 8 7 6 5 9 1 2]
  [6 9 7 2 1 4 8 3 5]
  [3 6 5 8 9 1 2 7 4]
  [8 7 2 4 5 3 6 9 1]
  [1 4 9 6 7 2 3 5 8]
  [7 8 4 1 3 6 5 2 9]
  [2 5 6 9 4 7 1 8 3]
  [9 1 3 5 2 8 4 6 7]]]
```

Time: 108 ms

# Backtracking



## Backtracking

- Algoritm
- Begränsningar

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

*GIF image from Wikipedia.com*

## Backtracking – Expert 04

- iterations: 427579
- Time: 545 ms

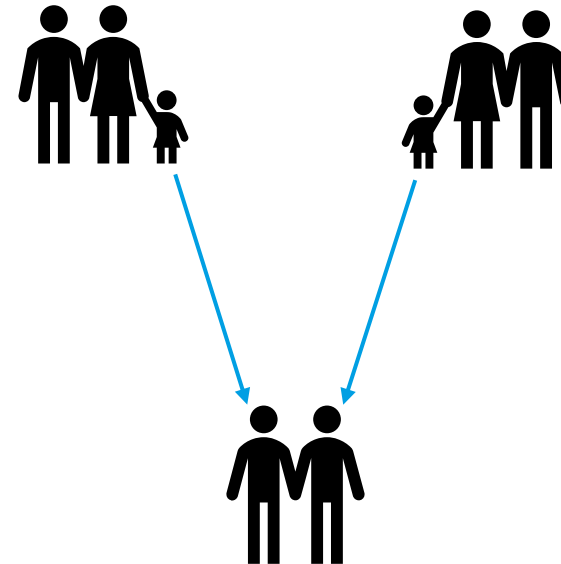
	5	2	1	3	8	9	7	4	6	
	4	3	8	7	6	5	9	1	2	
	6	9	7	2	1	4	8	3	5	
	3	6	5	8	9	1	2	7	4	
	8	7	2	4	5	3	6	9	1	
	1	4	9	6	7	2	3	5	8	
	7	8	4	1	3	6	5	2	9	
	2	5	6	9	4	7	1	8	3	
	9	1	3	5	2	8	4	6	7	

# GA



# GA

- Inspirerad från naturen
- Generator
- Fitness
- Vikt
- Korsare
- Mutation
- Tillstånd/Condition



## Vår GA

- Förinfyllda värden blir inte ändrade
- PMX
- Mutation rate
- Divisor
- Selection ratio
- Utan vikter
- Utan "grandparents"

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



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



## Ga – Resultat Easy 01

- Parametrar:
  - n\_parents=3000
  - n\_generations=10000
  - mutation\_rate=0.1
  - selection\_ratio=0.25
- PMX
  - 248 sekunder
  - 1075 generationer
- Row swap
  - 595 sekunder
  - 4811 generationer

```
[[6 1 3 5 4 9 2 8 7]
 [2 5 7 8 3 1 4 6 9]
 [9 8 4 6 7 2 3 5 1]
 [8 3 2 1 5 7 9 4 6]
 [7 4 5 3 9 6 1 2 8]
 [1 9 6 2 8 4 5 7 3]
 [3 7 8 4 1 5 6 9 2]
 [4 2 9 7 6 3 8 1 5]
 [5 6 1 9 2 8 7 3 4]]
```

# Jämförelser

# Lösningsmöjlighet & Komplexitet

# PuLP

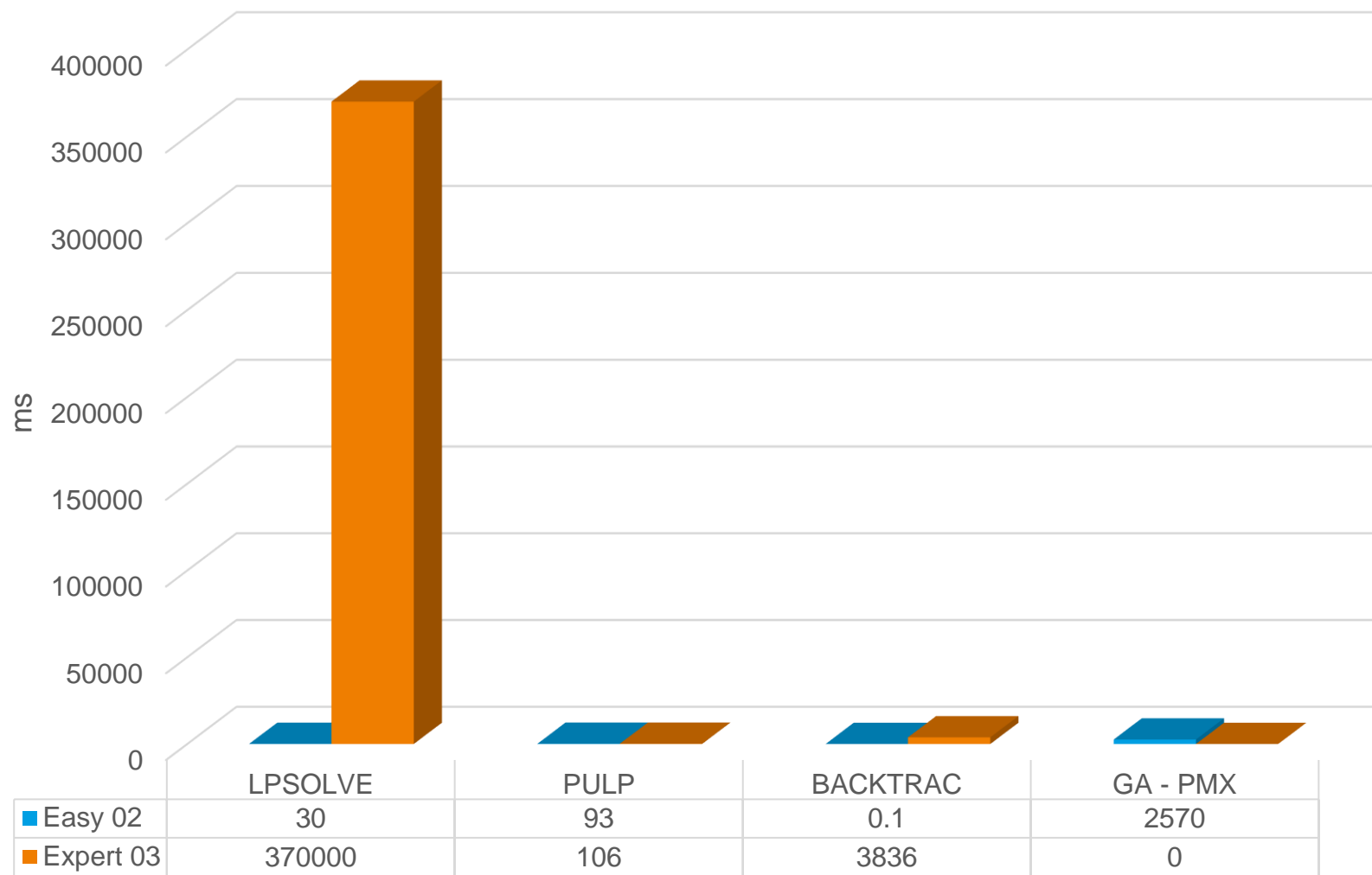
## Backtracking

GA

LPSOLVE

[illegible]

## Tid tills lösning



Samtliga sudokun tagna från [sudoku.com](https://www.sudoku.com)

---

**Tack för att ni lyssnade!**

