



SUDOKU SOLVER

**"SOLVING SUDOKU WITH ANT COLONY
OPTIMIZATION"**



"Intelligent Systems Incorporated"

Diego Ramos - 21951033

Lourdes Zamora - 21511351

Mario Flores - 21841245

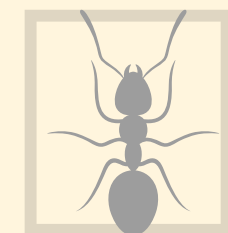
	6					5		2
	3				7			
	2		3		6			7
8	7	3		2	1	4	5	
9	4		5					
	1				4			
				9	5			4
3	9	4	8	1			7	5
	5	1		6	3		9	8

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



Por Qué?

Los algoritmos de propósito general de IA necesitan retos significativos para las pruebas de su funcionamiento, es por esto que se utiliza el sudoku solver como método de prueba para este caso una implementación de ACO.





REPRESENTACION DEL ENTORNO



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

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

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



REDUCCION DE ESTADOS

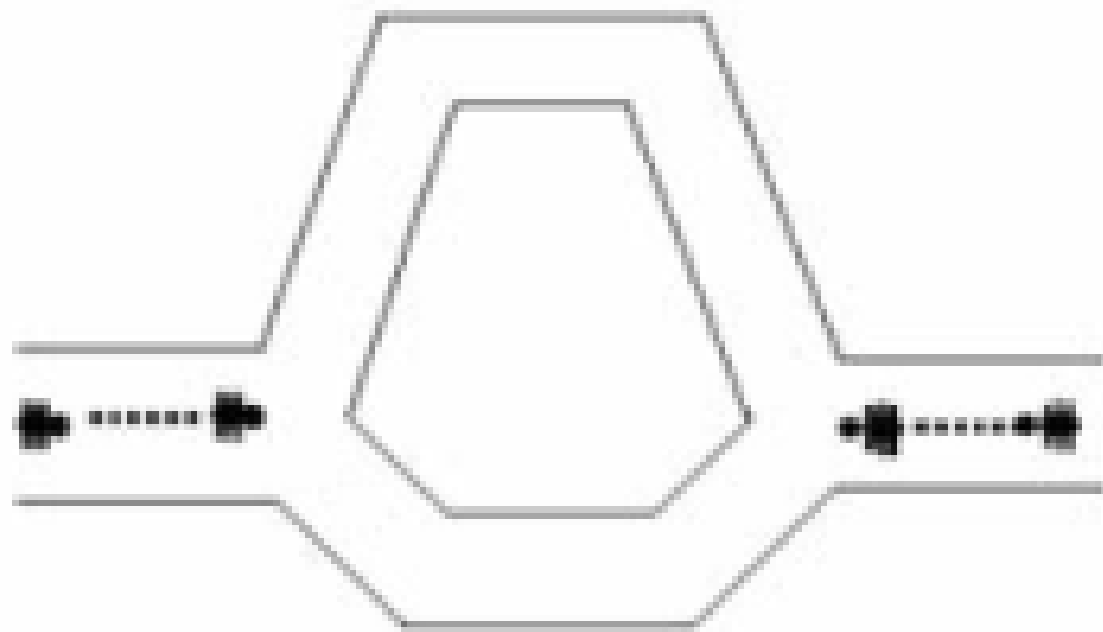


	6					5		2
	3				7			
	2		3		6			7
8	7	3		2	1	4	5	
9	4		5					
	1				4			
				9	5			4
3	9	4	8	1			7	5
	5	1		6	3		9	8

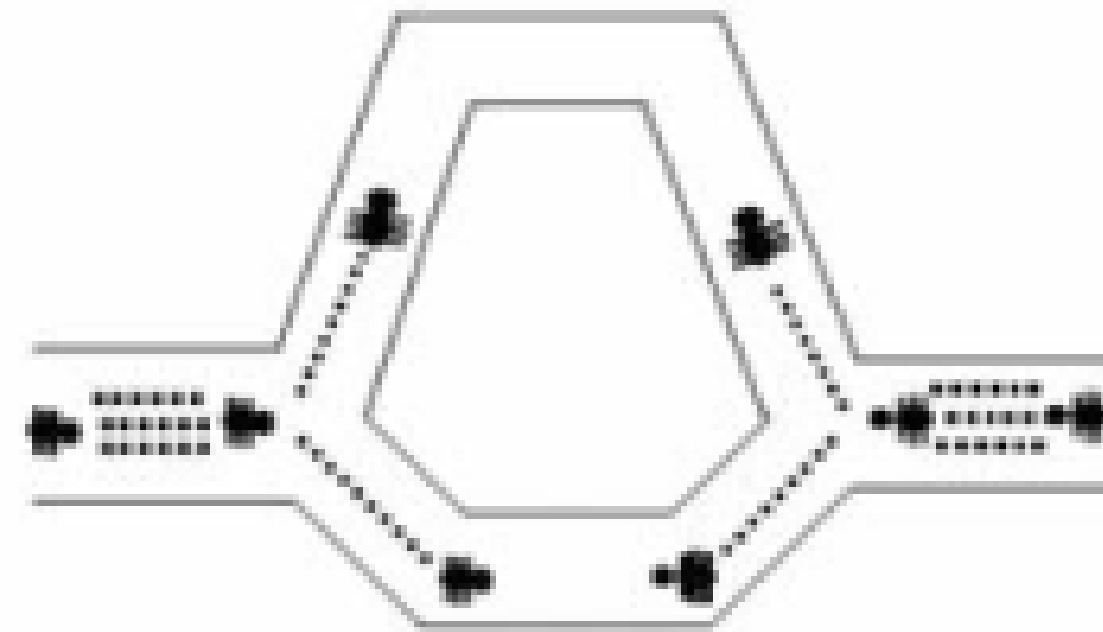
4	6	7		1	8	9		5	3	2
15	3	589		2	45	7		189	468	169
15	2	589		3	45	6		189	48	7
-----+-----+-----										
8	7	3		69	2	1		4	5	69
9	4	26		5	37	8		17	26	136
256	1	256		69	37	4		789	268	369
-----+-----+-----										
26	8	26		7	9	5		3	1	4
3	9	4		8	1	2		6	7	5
7	5	1		4	6	3		2	9	8

Ant Colony Optimization (ACO)

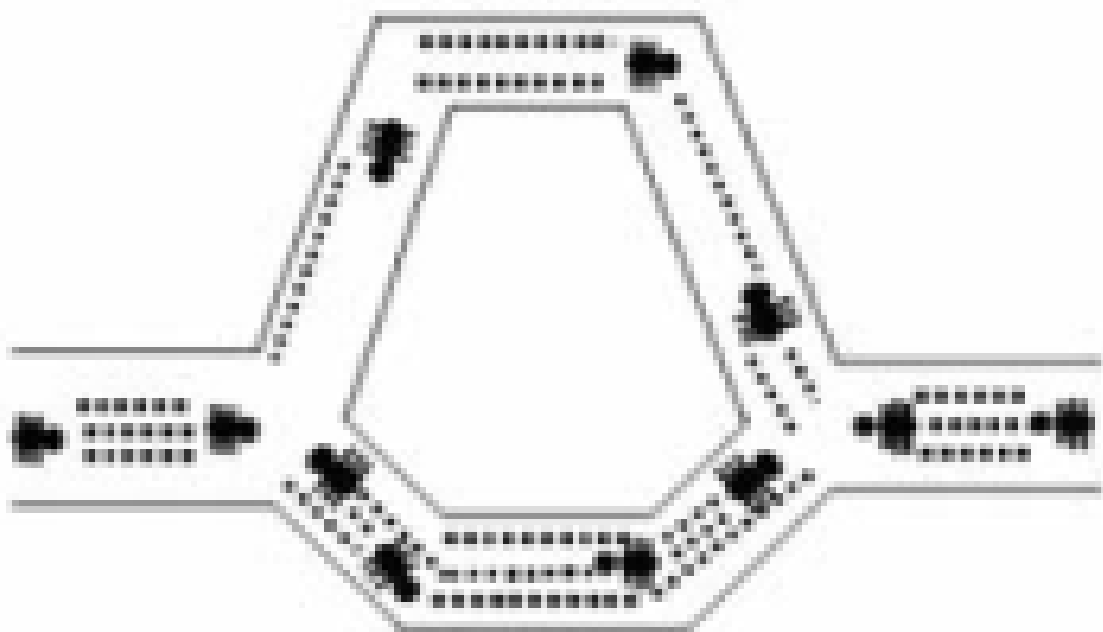
ACO es un metodo metaheuristico utilizado en algoritmos de busqueda. Es inspirado por el comportamiento de las hormigas. Un algoritmo ACO básico usa una población de hormigas que individualmente y en paralelo exploran el espacio de un problema y crean una solución incrementalmente combinado con una estructura de datos global que se utiliza para almacenar las decisiones tomadas por las hormigas.



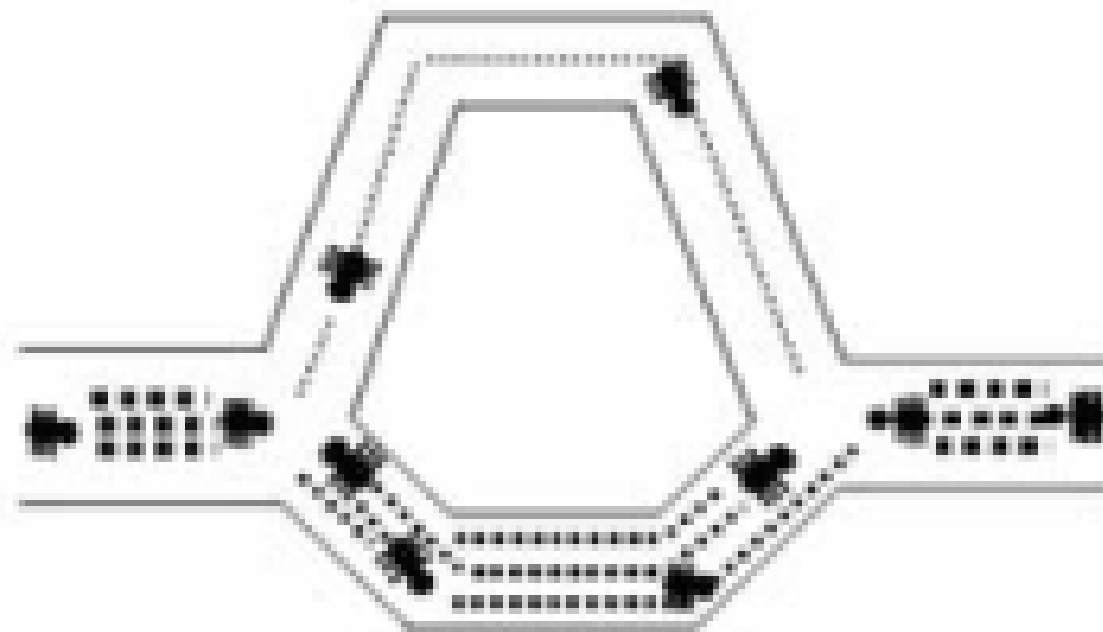
(a)



(b)



(c)



(d)

PseudoCodigo

Algorithm 1: Our ACO algorithm for Sudoku

```
1 read in puzzle;
2 for all cells with fixed values do
3   | propagate constraints (according to Section III-A);
4 end
5 initialize global pheromone matrix;
6 while puzzle is not solved do
7   | give each ant a local copy of puzzle;
8   | assign each ant to a different cell;
9   | for number of cells do
10    | | for each ant do
11    | | | if current cell value not fixed then
12    | | | | choose value from current cell's value set;
13    | | | | fix cell value;
14    | | | | propagate constraints;
15    | | | | update local pheromone;
16    | | | end
17    | | | move to next cell;
18    | | end
19   | end
20   | find best ant;
21   | do global pheromone update;
22   | do best value evaporation;
23 end
```



EXPERIMENTO



M

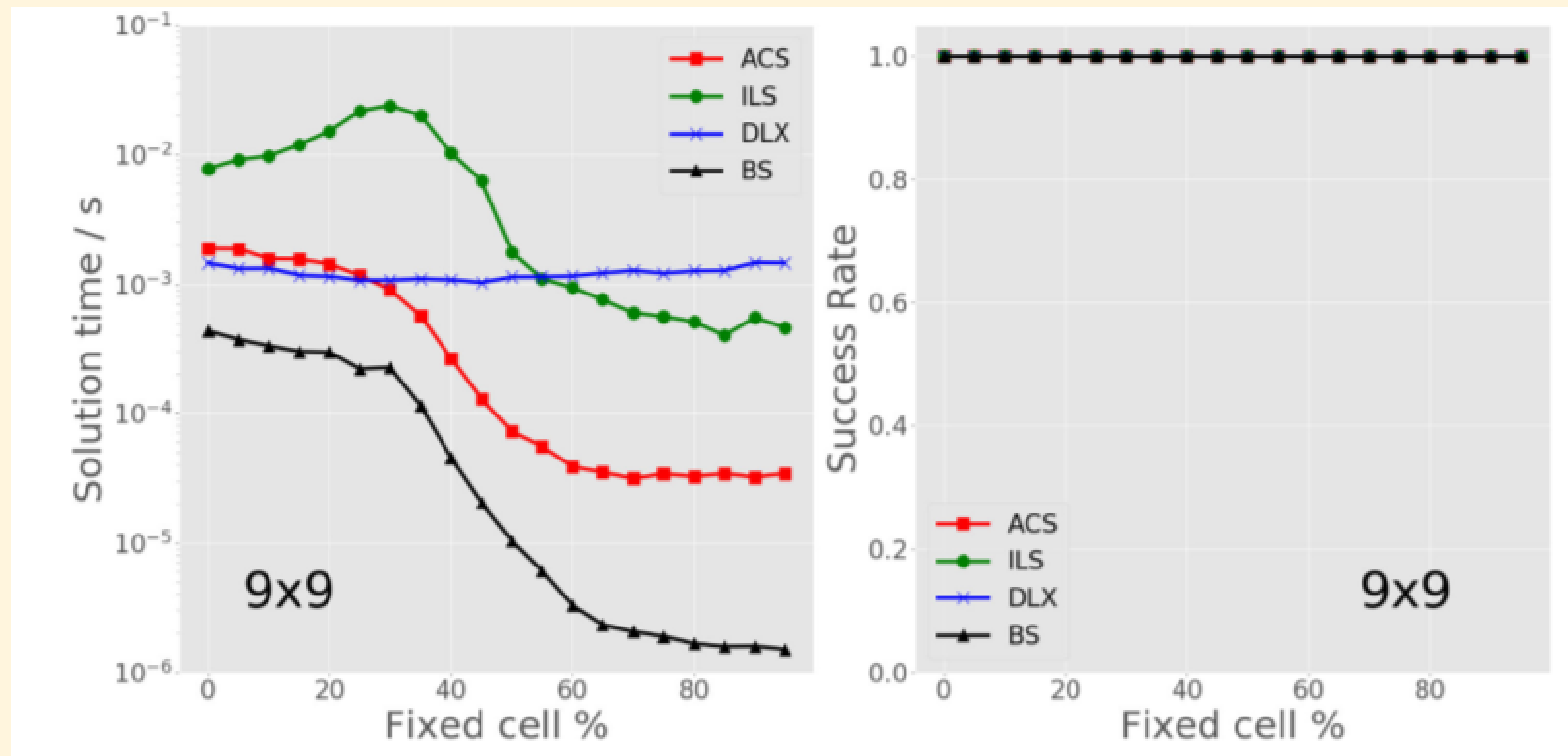
Descripción

- Se comparan los algoritmos ILS, DLX, BS, ACS (ACO adaptado).
- Se toman sudokus de 9x9 de la literatura investigada y, otros generados de manera aleatoria con dimensiones: 9x9 16x16 y 25x25.
- Se tomaron en cuenta el numero de instancias ,timeouts y el tiempo de resolución.



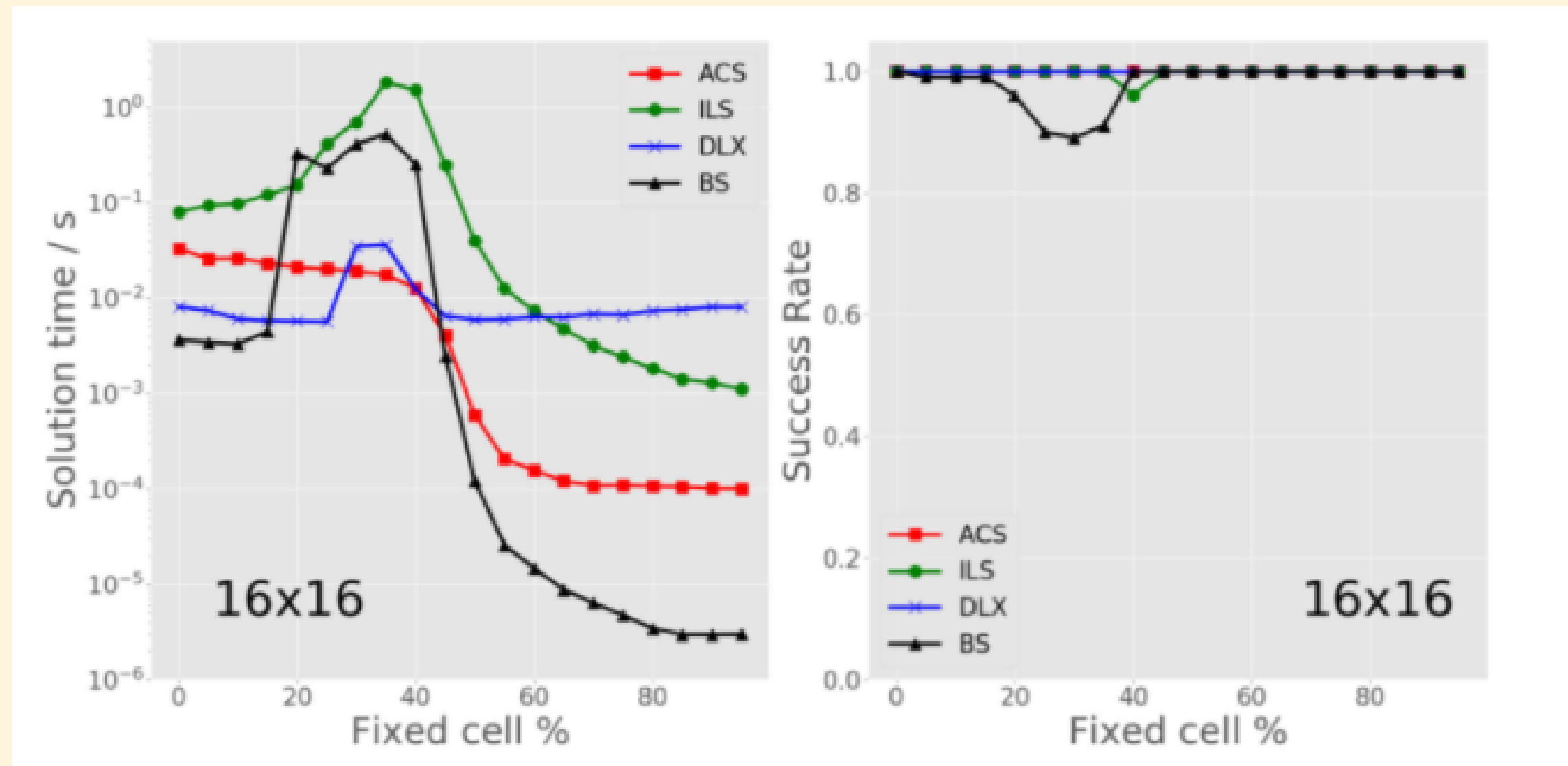
Resultados

Grafica de resultados para los ejemplos aleatorios de 9x9:



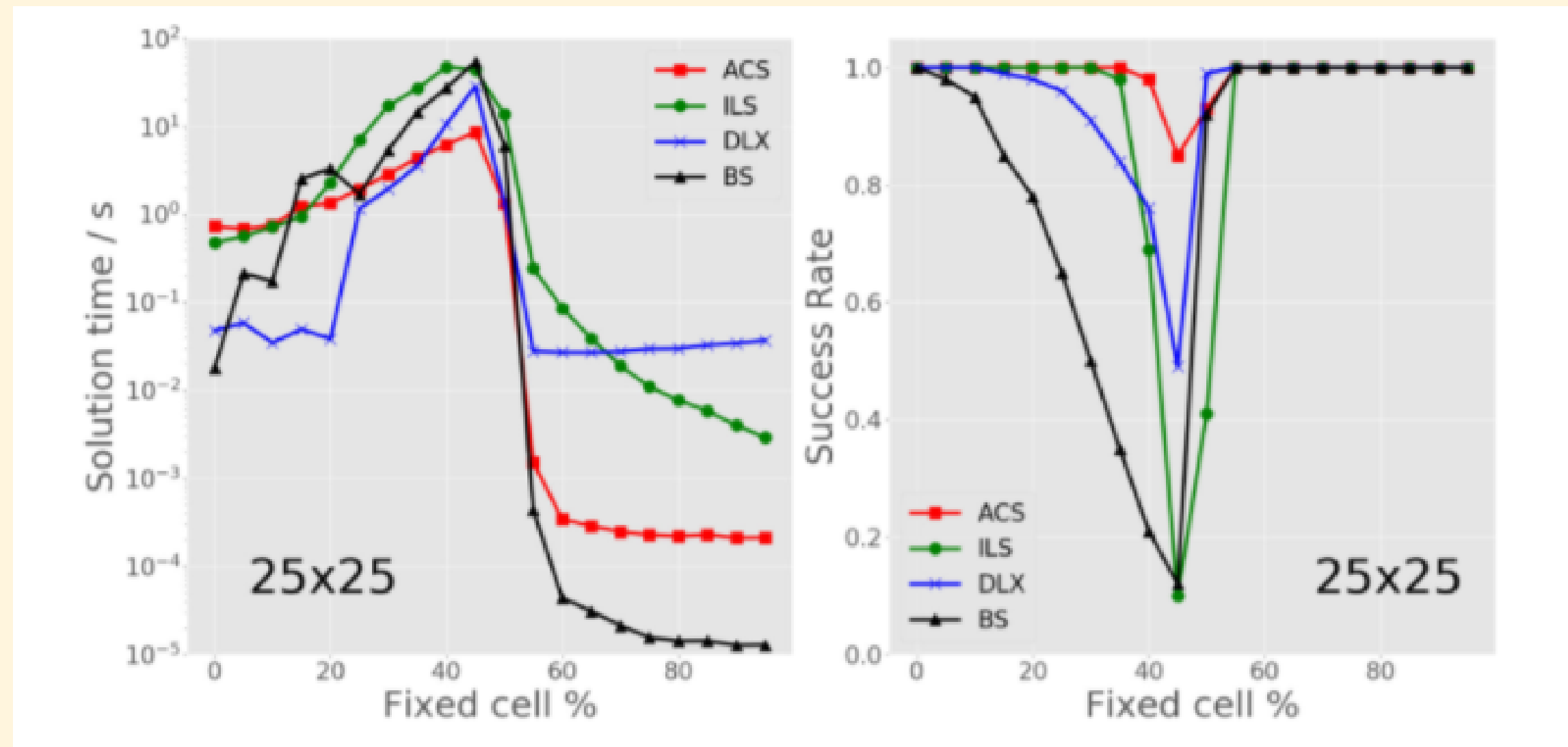
Resultados


Grafica de resultados para los ejemplos aleatorios de 16x16:




Resultados

Grafica de resultados para los ejemplos aleatorios de 25x25:



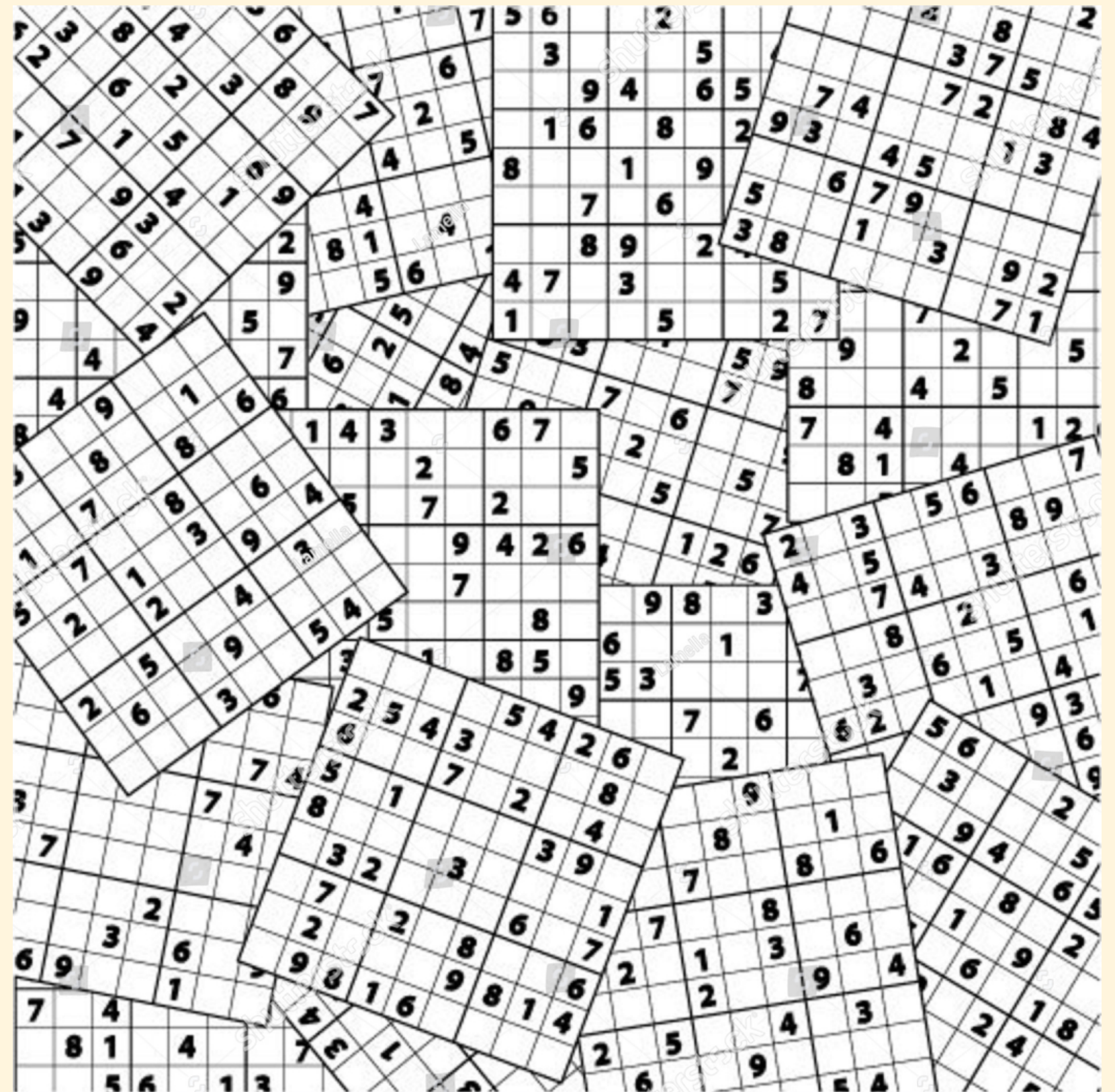


**Posibles
aplicaciones para
el proyecto.**



Objetivos del Prototipo

- Generar configuraciones validas de Sudoku.
- Crear un algoritmo básico que pueda resolver Sudokus regulares.
- Permitir crear y resolver Sudokus de diferentes tipos.
- Crear una visualización de cómo se resuelve el sudoku paso a paso.



https://github.com/huwlloyd-mmu/sudoku_acs



CONCLUSIONES





GRACIAS

