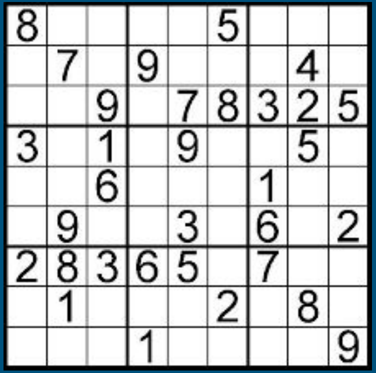
Sudoku Solver

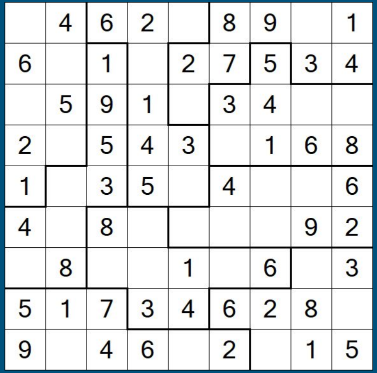
Lyn Yandik, Varun Mahesh

Problem Description

- Our task was to use a Constraint Satisfaction Problem (CSP) to solve a sudoku
- Sudoku is a puzzle in which every row, column, and 3 by 3 box in the 9 by 9 grid must contain every digit 1-9 with no repeats
- Additionally, we have also solved variants with different box shapes, known as "region" or "jigsaw" sudokus



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



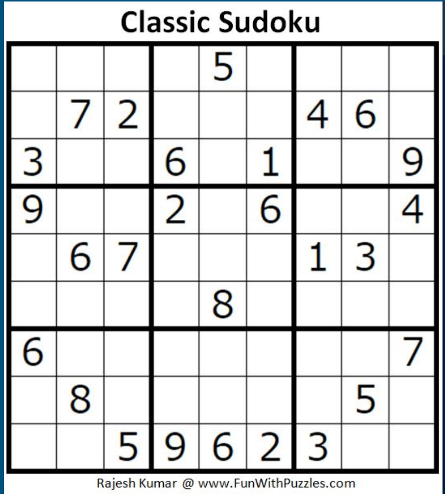
	4	6	2		8	9		1
6		1		2	7	5	3	4
	5	9	1		3	4		
2		5	4	3		1	6	8
1		3	5		4			6
4		8					9	2
	8			1		6		3
5	1	7	3	4	6	2	8	
9		4	6		2		1	5

Problem Description

- Additionally, our project also involved examining the changes in speed and efficiency between using plain backtracking CSP and CSP enhanced with the AC3 algorithm
- Our implementation of these was based on our work for Assignment 2, descriptions given in class, and the GeeksForGeeks example implementation of a CSP and AC3

Data Set

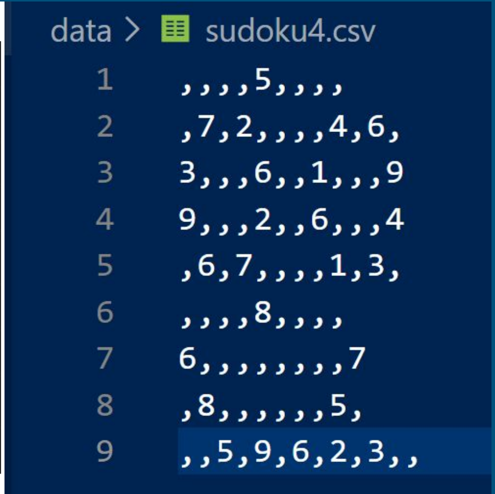
- Our testing data consisted of free sudoku puzzles found online for free, with a range of labeled difficulty levels and amount of given numbers
- We typed these puzzles into .csv files of nine rows and nine columns for processing in our program
  - Existing digits were in their positions
  - Empty positions were left blank
- We included regional sudoku
  - Boxes are replaced with new areas
  - Area input is added to the csv file



Classic Sudoku

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

Rajesh Kumar @ www.FunWithPuzzles.com



data > sudoku4.csv

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

Backtracking Algorithm

The backtracking algorithm sets a domain for each cell and assigns variables to every possible permutation until it finds a valid solution. Without AC3, the domain for each cell is 0-9 except for set cells with only 1 possible value.

When assigning unfilled cells, we prioritize the MRV (Minimum Remaining Value) instead of the LCV (Least Constraining Value) because the LCV chooses cells depending on the amount of neighbors, but every cell has the exact same amount of neighbors.

AC3 Algorithm

The AC3 algorithm extends from the backtracking solver and has an additional method to reduce the domain of all cells in the sudoku puzzle. The variables are represented by each cell, the arcs are cells that neighbor a cell in either of 3 groups (horizontal, vertical, box). All constraints for the arcs are the same, being that both cells in the arc are not equal to each other.

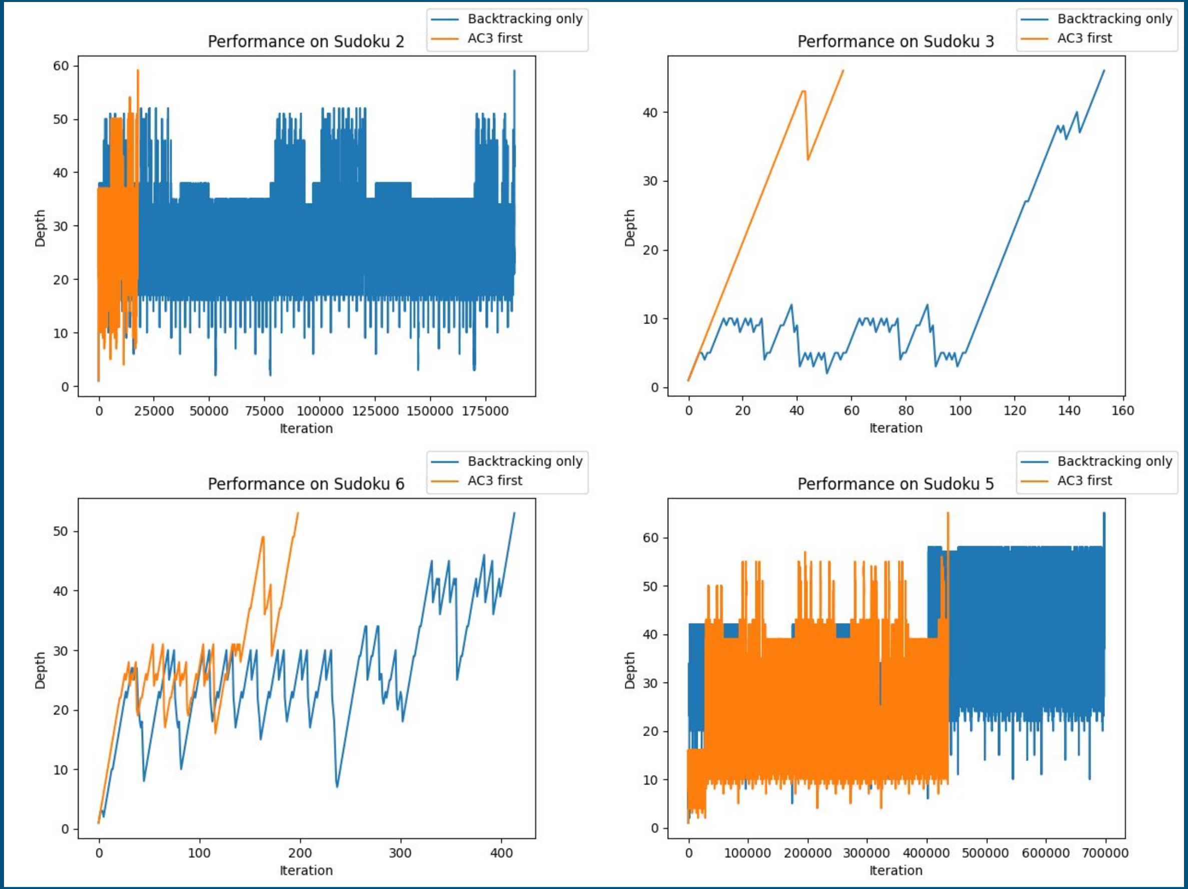
When the algorithm is finished, the domains are reduced to only include numbers in valid solutions.

Evaluation Method

- We recorded backtracking depth at each run of the loop
- These were turned into graphs with matplotlib comparing plain backtracking and backtracking when AC3 was run first
- We ran the algorithm on 8 regular sudokus and 2 region variant sudokus for 10 total results

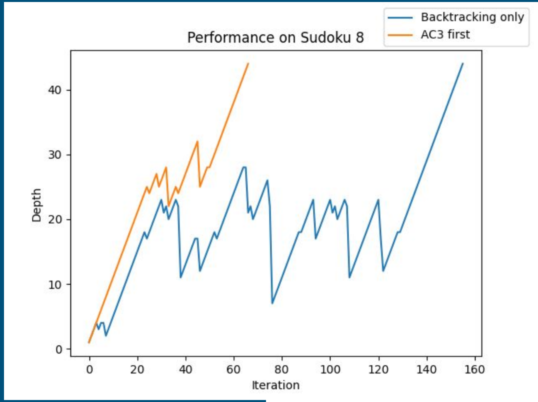
Results

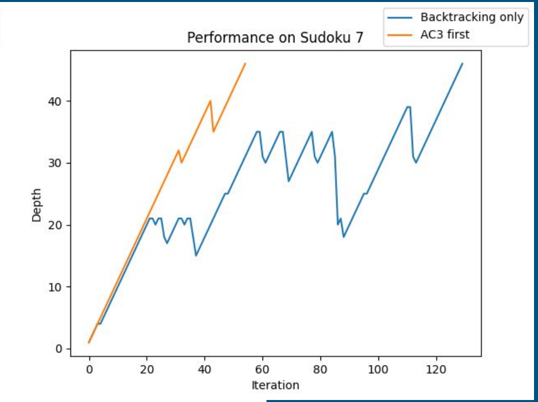
- Generally, running AC3 allowed the algorithm to perform much better than backtracking alone
- Improvement ranged from extremely significant to more minor
    - Shown are 4 examples demonstrating both significant and minor improvements

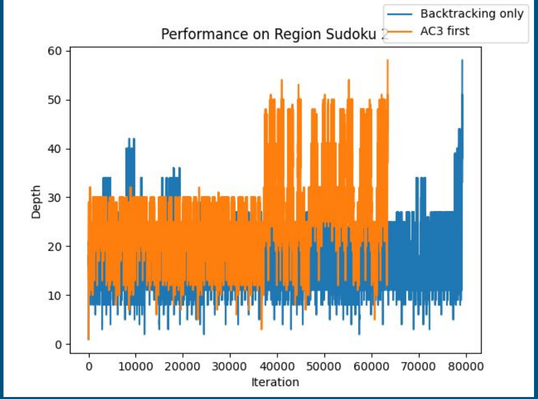


Why is AC3 Better?

- With backtracking only, AC3 has a smaller domain and therefore it contains less permutations for nearly all sudoku puzzles
- There was less branching in configurations because of the smaller domains
- Given the sheer size of the possible amount of configurations, the runtime will have significant change

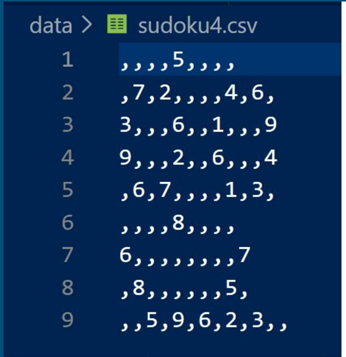






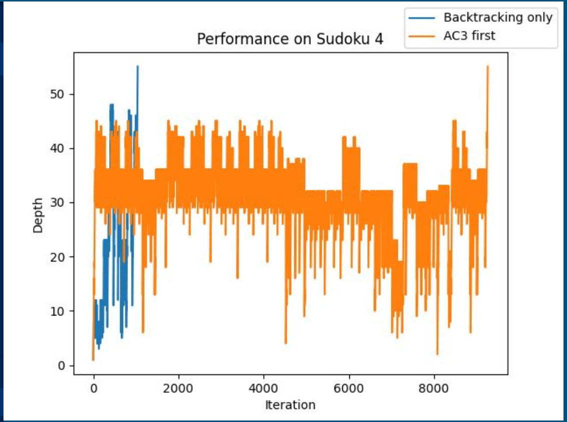
Results: Outliers

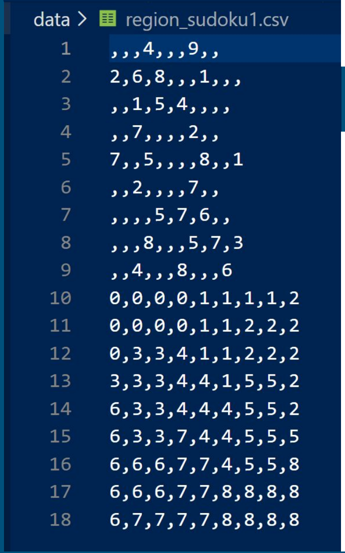
- However, there were several instances in which AC3 actually performed *worse* than backtracking alone
- These cases were sudoku 1, sudoku 4, and region sudoku 1



data > sudoku4.csv

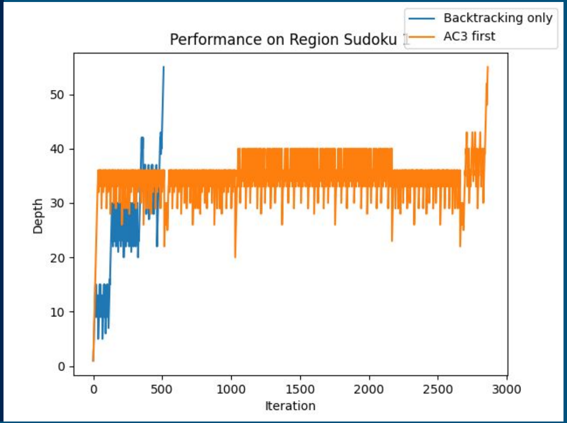
```
,,,,5,,,,
,7,2,,,,4,6,
3,,,6,,1,,,9
9,,,2,,6,,,4
,6,7,,,,1,3,
,,,,8,,,,
6,,,,,,,,7
,8,,,,,,5,
,,5,9,6,2,3,,
```





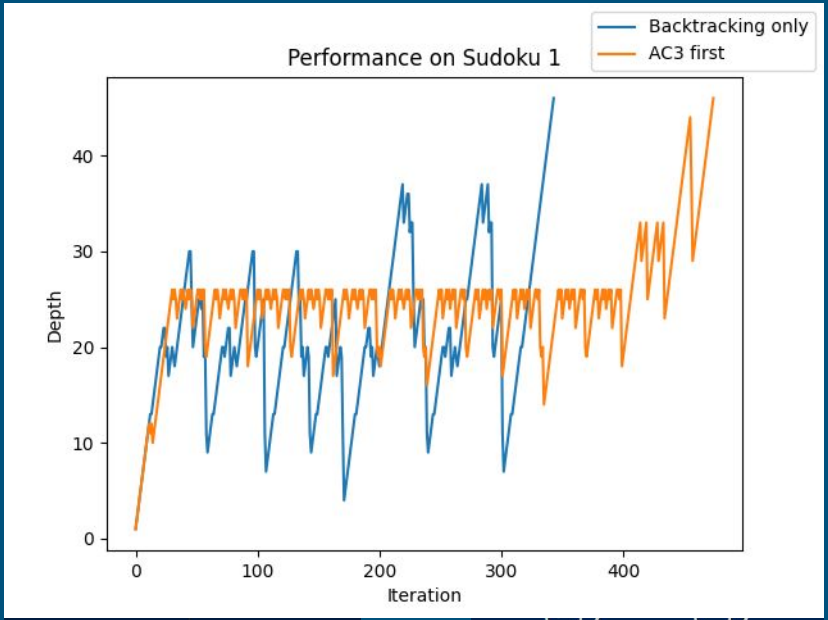
data > region_sudoku1.csv

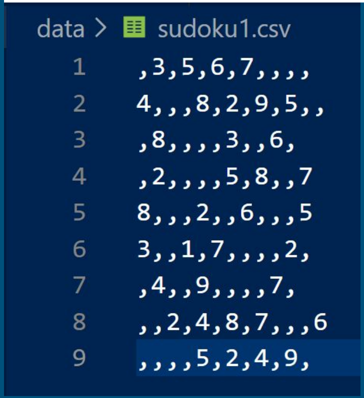
```
,,,4,,,9,,
2,6,8,,,,1,,,
,,1,5,4,,,,
,,7,,,,2,,
7,,5,,,,8,,1
,,2,,,,7,,
,,,,5,7,6,,
,,,8,,,5,7,3
,,4,,,8,,,6
0,0,0,0,1,1,1,1,2
0,0,0,0,1,1,2,2,2
0,3,3,4,1,1,2,2,2
3,3,3,4,4,1,5,5,2
6,3,3,4,4,4,5,5,2
6,3,3,7,4,4,5,5,5
6,6,6,7,7,4,5,5,8
6,6,6,7,7,8,8,8,8
6,7,7,7,7,8,8,8,8
```



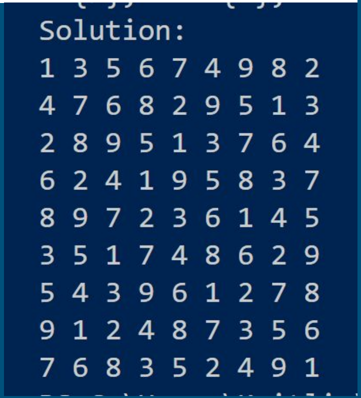
Why Could This Be?

- We are prioritizing nodes with least possible options
  - In regular backtracking, once it runs through the given values, it simply goes in order
  - The method of selecting values in a pattern may cause backtracking alone to find potential mistakes and problems faster as it checks several neighbors in a row
  - AC3 causes a jump around that may prevent it from catching early mistakes by not testing that value's neighbors as quickly
- Differences in sudoku design
  - It's possible that certain sudokus have patterns that cause AC3 to guess wrong early or get stuck




data > sudoku1.csv

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


```
Solution:
1 3 5 6 7 4 9 8 2
4 7 6 8 2 9 5 1 3
2 8 9 5 1 3 7 6 4
6 2 4 1 9 5 8 3 7
8 9 7 2 3 6 1 4 5
3 5 1 7 4 8 6 2 9
5 4 3 9 6 1 2 7 8
9 1 2 4 8 7 3 5 6
7 6 8 3 5 2 4 9 1
```

Could We Do Better?

- With forward checking, we could potentially further improve the algorithm, including potentially preventing or at least improving the enhanced algorithm outlier cases
- Additional domain updates during the backtracking run could also improve selections
  - One idea I had was to remove the values assigned to already defined positions from the domains of their neighbors in the initial constructor
  - However, since the algorithm only proceeds with valid constructions anyway, this could only potentially cause a small speed increase rather than depth improvements
  - It also seemed against the spirit of plain backtracking and AC3 removes them anyway

Conclusions

- Overall, using AC3 causes an increase in efficiency and speed when solving CSPs
- AC3's "jumping" with unassigned variables means it can sometimes be slower than backtracking
  - This is because of the method in which variables are selected, not because AC3 itself makes the algorithm slower

References

- https://www.geeksforgeeks.org/artificial-intelligence/constraint-propagation-in-ai/