TD1: Constraints & Data Mining

Pablo Mollá Chárlez

January 17, 2025

Contents

1 Exercises

1.1 Exercise 1

Consider the following addition problem:

```
  SEND
+ MORE
-------
= MONEY
```

Figure 1: Addition Problem

Each letter represents a distinct digit between 0 and 9. We need to determine the value of each letter, keeping in mind that the first letter of each word is different from zero.

1.2 Exercise 2

Consider a chessboard of size $(N \times N)$. The N-Queens problem consists of placing N queens such that none of them can attack each other.

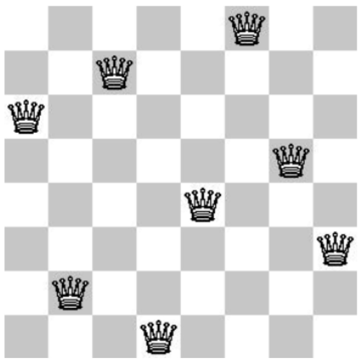


Figure 2: 8-Queens Problem Solved

- **Question 1:** Model the problem as a constraint satisfaction problem (CSP) $N = \langle X, D, C \rangle$.
- **Question 2:** What is the size of the search space for this problem on a $N \times N$ chessboard?
- **Question 3:** Run the Backtracking (BT), Forward Checking (FC), and Maintaining Arc Consistency (MAC) algorithms on the partial instantiation given bellow.

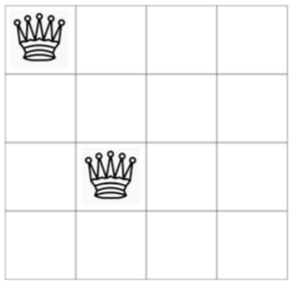


Figure 3: Partial Instantiation of 4-Queens Problem

1.3 Exercise 3

Consider the constraint network described below:

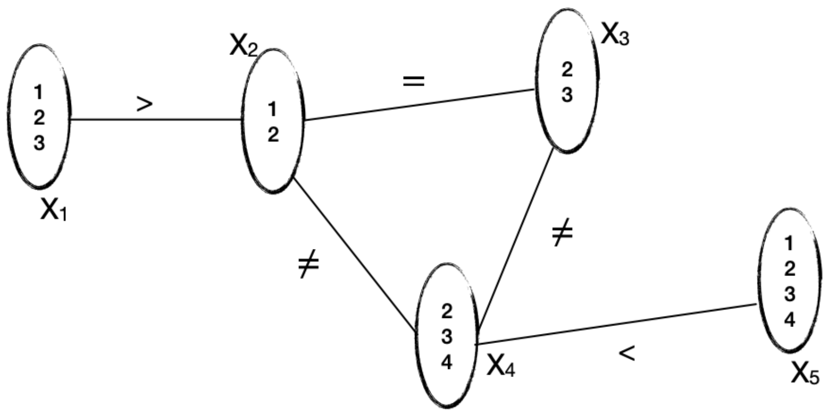


Figure 4: Constraint Network

- **Question 1** Write the initial propagation queue (list of arcs) that AC3 will start with. Remember to include both directions for each constraint. For example, for $x_1 \neq x_2$, include $\overrightarrow{x_1 \neq x_2}$ and $\overleftarrow{x_1 \neq x_2}$.
- **Question 2** Using the $AC-3$ algorithm, manually process the propagation queue step by step. For each step:
  - Indicate which arc is being checked.
  - Show any changes made to the domains of the variables.
  - If a domain changes, add the necessary arcs back to the queue.
- **Question 3** At the end of the process, write the final domains of all variables.
- **Question 4** Was the constraint network arc-consistent after running $AC-3$? Explain briefly.

1.4 Exercise 4

A Golomb ruler is a rule that contains marks at integer positions such that every pair of marks has a different length between them.

- **Question 1:** Model the problem as a constraint network $N =, D, C\rangle$.
- **Question 2:** Provide the optimization version of the problem, with the objective of returning the smallest Golomb ruler.
- **Question 3:** In pairs, conduct a comparative study of the different versions of the model, from the basic version **GolombRuler1.java** to the most refined and optimized version **GolombRuler5.java**. Write a brief report presenting the improvements made by each refinement and the gains achieved in terms of performance or accuracy.
- **Question 4:** In pairs, analyze whether the constraint $tiks[m-1] \geq m(m-1)/2$ can be added to the model. Discuss whether it is beneficial to include it or not, and justify your answer in a brief report.

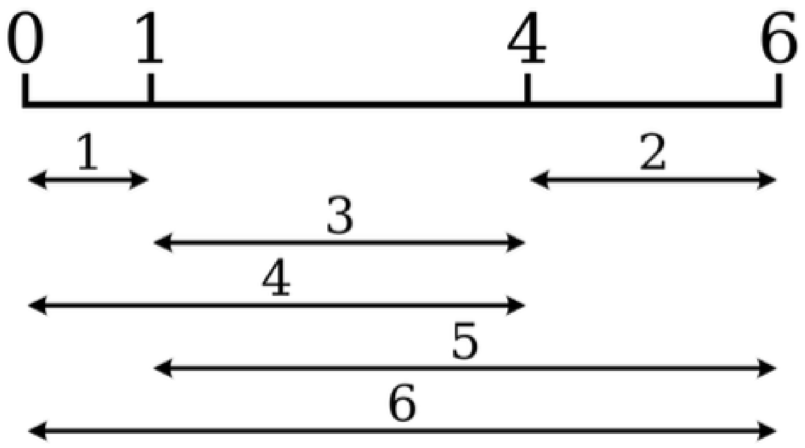
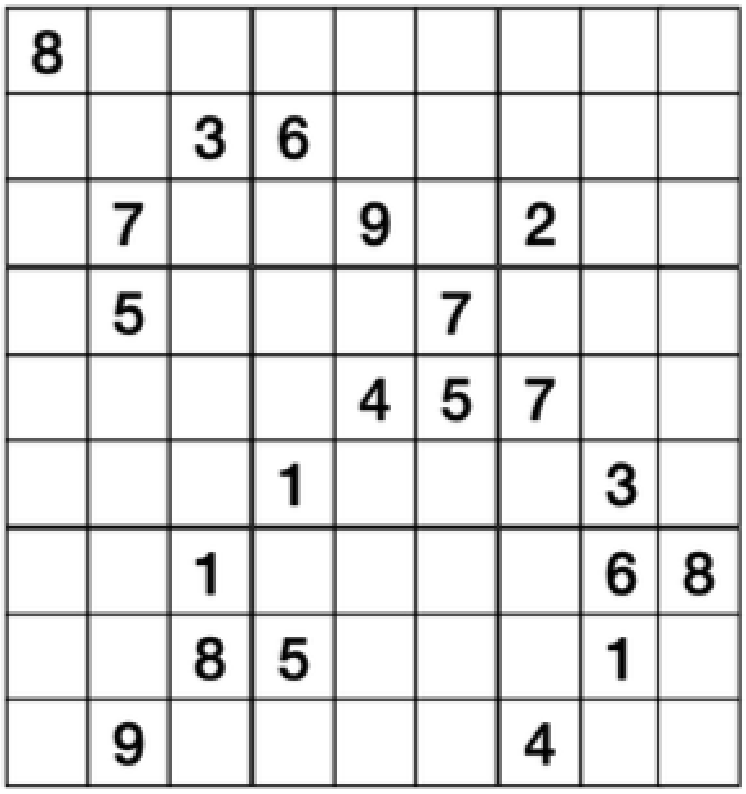


Figure 5: Golomb Ruler

1.5 Exercise 5

This assignment is to be completed in pairs, and the report must be submitted jointly.

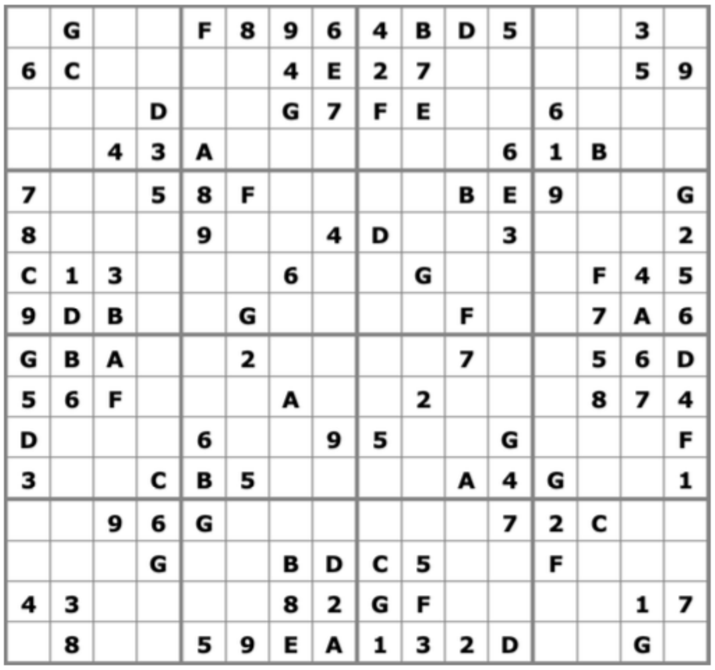
- **Question 1:** Model the Sudoku problem on paper as a Constraint Network N= ⟨X, D, C⟩. You will find in your local repository the file Sudoku.java, which contains a Constraint Programming (CP) model of the Sudoku problem using the Choco-Solver library.
- **Question 2:** Modify the provided code to return all possible solutions to the Sudoku puzzle. Next, we will test the declarative nature of CP by applying modifications to the provided model. Figure 6 depicts one of the most challenging instances of a 9 ×9 Sudoku puzzle.



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

Figure 6: Difficult instance of 9×9 Sudoku

- **Question 3:** Modify the CP model in Sudoku.java to solve the instance shown in Figure 7.



	G			F	8	9	6	4	B	D	5			3	
6	C					4	E	2	7					5	9
			D			G	7	F	E			6			
		4	3	A							6	1	B		
7			5	8	F					B	E	9			G
8				9			4	D			3				2
C	1	3				6			G				F	4	5
9	D	B			G					F			7	A	6
G	B	A			2					7			5	6	D
5	6	F				A			2				8	7	4
D				6			9	5			G				F
3			C	B	5					A	4	G			1
		9	6	G							7	2	C		
			G			B	D	C	5			F			
4	3					8	2	G	F					1	7
	8			5	9	E	A	1	3	2	D			G	

Figure 7: Instance of 16 × 16 Sudoku

- **Question 4:** Adapt your code to handle both the instance in Figure 6 and the one in Figure 7.
- **Question 5:** One variation of the classic Sudoku is the Greater Than Sudoku (GTSudoku), an example of which is shown in Figure 3. In addition to the constraints of the classic Sudoku, GTSudoku introduces comparison symbols (> and <) in the grid. These symbols indicate inequality constraints between adjacent cells within the same sub-grid. Revise the model in **Sudoku.java** to solve the GTSudoku instance in Figure 8.

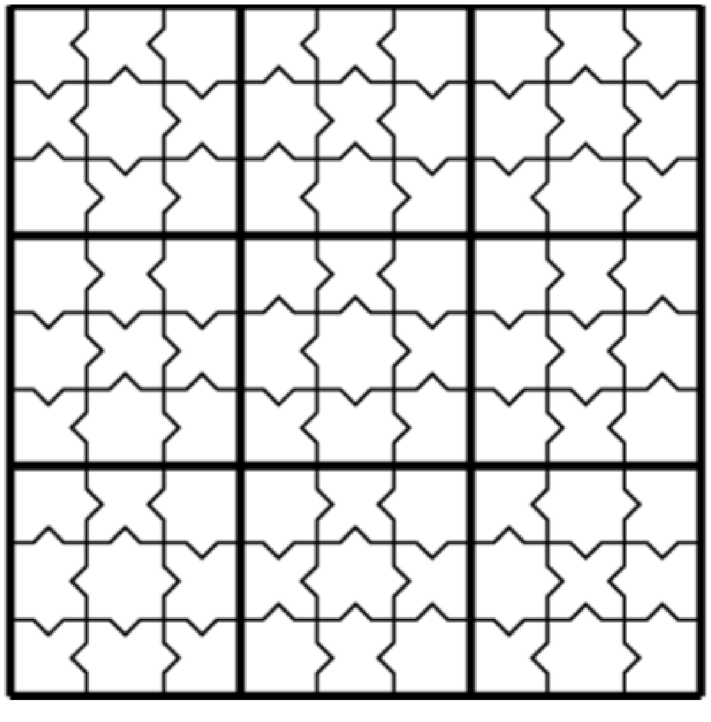


Figure 8: Instance of GT-Sudoku (9 × 9

2 Solutions

2.1 Exercise 1

Let's model the problem in 2 different ways.

- **1st Model Approach:** Without Auxiliary Variables
  - The variables are defined as: $X = \{S, E, N, D, M, O, R, Y\}$.
  - The domain of those variables is: $D = d^8$ where $d = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.
  - The constraints are defined as: $C = \{S \neq 0, M \neq 0,$
    All variables are different $\equiv \forall (x_i, x_j) \in X^2$ such that $i < j$ then $x_i \neq x_j$,
    Equation $\equiv \overbrace{1000S + 100E + 10N + D}^{SEND} + \overbrace{1000M + 100O + 10R + E}^{MORE} = \overbrace{10000M + 1000O + 100N + 10E + Y}^{MONEY}\ \}$
- **2nd Model Approach:** Adding Auxiliary Variables
  - The variables are defined as: $X = \{S, E, N, D, M, O, R, Y, R_1, R_2, R_3\} = \{S, E, N, D, M, O, R, Y\} \cup \{R_1, R_2, R_3\} = X' \cup X''$.
  - The domain of those variables is: $\forall x_i \in X'$ where $D(x_i) = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and $\forall x_i \in X'', D(x_i) = \{0, 1\}$.
  - The constraints are defined as: $C = \{S \neq 0, M \neq 0,$
    All variables are different $\equiv \forall (x_i, x_j) \in X' \times X'$ such that $i < j$ then $x_i \neq x_j$,
    Equation $\equiv$
    $$D + E = Y + 10R_1, \quad N + R + R_1 = E + 10R_2, \quad E + O + R_2 = N + 10R_3, \quad S + M + R_3 = O + 10M$$

2.2 Exercise 2

The problem can be modeled as follows:

- The variables are defined as: $X = \{C_1, \ldots, C_N\}$ where $C_i = a$ represents that in row i there is a queen in column a.
- The domain of those variables is $D = \{1, \ldots, N\}$, therefore the search space: $|D|^{|X|} = N^N$.
- The constraints are defined as: $C = \{$Column Constraints $\cup$ Diagonal Constraints $\}$
  $= \{\forall i, j \in \{1, \ldots, N\}$ such that $i < j,\ C_i \neq C_j \ \cup\ \forall i, j \in \{1, \ldots, N\}$ such that $i < j,\ |C_i - C_j| \neq |i - j|\}$

For this course, we won't need to remember the algorithms Backtracking (BT) and Forward-Checking (FC), however Maintaining Arc Consistency (MAC) is required, so let's proceed then starting from the information we know, $C_1 = 1$ and $C_2 = 3$. We will first consider the possible options for C_3. As we know, $C_3 \neq C_i \forall i$, therefore either $C_3 = 2$ or $C_3 = 4$. Besides, $\forall i \in \{1, 2\}$ such that $3 < i$, $|C_3 - C_i| \neq |3 - i|$. Then,

- **Case $C_3 = 2$:**
  - $|C_3 - C_1| = |2 - 1| = 1 \neq 2 = |3 - 1| = |3 - i|$ ✓
  - $|C_3 - C_2| = |2 - 3| = 1 \neq 1 = |3 - 2| = |3 - i|$ ✗

  However the condition is not satisfied $\forall i \in \{1, 2\}$.
- **Case $C_3 = 4$:**
  - $|C_3 - C_1| = |4 - 1| = 3 \neq 2 = |3 - 1| = |3 - i|$ ✓

  - $|C_3 - C_2| = |4 - 3| = 1 \neq 1 = |3 - 2| = |3 - i|$ ✗

  However the condition is not satisfied $\forall i \in \{1, 2\}$.

Therefore, no possible value can be assigned to C_3. We continue with C_4, which we know needs to satisfy that $C_4 \neq C_1$ and $C_4 \neq C_2$, therefore either $C_4 = 2$ or $C_4 = 4$. As previously:

- **Case $C_4 = 2$:**
  - $|C_4 - C_1| = |2 - 1| = 1 \neq 3 = |4 - 1| = |4 - i|$ ✓
  - $|C_4 - C_2| = |2 - 3| = 0 \neq 2 = |4 - 2| = |4 - i|$ ✓

  All conditions are satisfied, we conclude that $C_4 = 2$.

- **Case $C_4 = 4$:**
  - $|C_4 - C_1| = |4 - 1| = 3 \neq 3 = |4 - 1| = |4 - i|$ ✗
  - $|C_4 - C_2| = |4 - 3| = 1 \neq 2 = |4 - 2| = |4 - i|$ ✓

  However the condition is not satisfied $\forall i \in \{1, 2\}$.

2.3 Exercise 3

The initial propagation queue is defined as follows:

$$Q = \{\overrightarrow{x_1 > x_2}, \overrightarrow{x_2 < x_1}, \overrightarrow{x_2 = x_3}, \overrightarrow{x_3 = x_2}, \overrightarrow{x_2 \neq x_4}, \overrightarrow{x_4 \neq x_2}, \overrightarrow{x_3 \neq x_4}, \overrightarrow{x_4 \neq x_3}, \overrightarrow{x_4 < x_5}, \overrightarrow{x_5 > x_4}\}$$

Let's proceed with the algorithm $AC - 3$ and remind that the orignal domains of each variable are defined within the constraint network:

1. $\overrightarrow{x_1 > x_2}$ $d_{x_1} = \{1, 2, 3\}$ & $d_{x_2} = \{1, 2\}$
   - $x_1 = 1 > 1$ ✗
   - $x_1 = 2 > 1$ ✓
   - $x_1 = 3 > 1$ ✓

   Then, $d_{x_1} = \{2, 3\} \implies Q = Q \cup \{\text{All arcs that reach } x_1\}$. (since we have started no need)

2. $\overrightarrow{x_2 < x_1}$ $d_{x_2} = \{1, 2\}$ & $d_{x_1} = \{2, 3\}$
   - $x_2 = 1 < 2$ ✓
   - $x_2 = 2 < 3$ ✓

   Then, no changes.

3. $\overrightarrow{x_2 = x_3}$ $d_{x_2} = \{1, 2\}$ & $d_{x_3} = \{2, 3\}$
   - $x_2 = 1 \neq 2, 3$ ✗
   - $x_2 = 2 = 2$ ✓

   Then, $d_{x_2} = \{2\} \implies Q = Q \cup \{\overrightarrow{x_1 > x_2}\}$.

4. $\overrightarrow{x_3 = x_2}$ $d_{x_2} = \{2\}$ & $d_{x_3} = \{2, 3\}$
   - $x_3 = 2 = 2$ ✓
   - $x_3 = 3 \neq 2$ ✗

Then, $d_{x_3} = \{2\} \implies Q = Q \cup \{\overrightarrow{x_2 = x_3}\}$.

5. $\overrightarrow{x_2 \neq x_4}$ $d_{x_2} = \{2\}$ & $d_{x_4} = \{2,3,4\}$

- $x_2 = 2 \neq 3$ ✓

Then, no changes.

6. $\overrightarrow{x_4 \neq x_2}$ $d_{x_2} = \{2\}$ & $d_{x_4} = \{2,3,4\}$

- $x_4 = 2 = 2$ ✗
- $x_4 = 3 \neq 2$ ✓
- $x_4 = 4 \neq 3$ ✓

Then, $d_{x_4} = \{3,4\} \implies Q = Q \cup \{\overrightarrow{x_2 \neq x_4}\}$.

7. $\overrightarrow{x_3 \neq x_4}$ $d_{x_3} = \{2\}$ & $d_{x_4} = \{3,4\}$

- $x_3 = 2 \neq 3,4$ ✓

Then, no changes.

8. $\overrightarrow{x_4 \neq x_3}$ $d_{x_3} = \{2\}$ & $d_{x_4} = \{3,4\}$

- $x_4 = 3 \neq 2$ ✓
- $x_4 = 4 \neq 2$ ✓

Then, no changes.

9. $\overrightarrow{x_4 < x_5}$ $d_{x_4} = \{3,4\}$ & $d_{x_5} = \{1,2,3,4\}$

- $x_4 = 3 < 4$ ✓
- $x_4 = 4 < 4$ ✗

Then, $d_{x_4} = \{3\} \implies Q = Q \cup \{\overrightarrow{x_3 \neq x_4}, \overrightarrow{x_2 \neq x_4}\}$.

10. $\overrightarrow{x_5 > x_4}$ $d_{x_4} = \{3\}$ & $d_{x_5} = \{1,2,3,4\}$

- $x_5 = 1 > 3$ ✗
- $x_5 = 2 > 3$ ✗
- $x_5 = 3 > 3$ ✗
- $x_5 = 4 > 3$ ✓

Then, $d_{x_5} = \{4\} \implies Q = Q \cup \{\overrightarrow{x_4 < x_5}\}$.

Now we start with the arcs that have been added to the queue since the beginning of the algorithm.

11. $\overrightarrow{x_1 > x_2}$ $d_{x_1} = \{2,3\}$ & $d_{x_2} = \{2\}$

- $x_1 = 2 > 2$ ✗
- $x_1 = 3 > 2$ ✓

Then, $d_{x_1} = \{3\} \implies Q = Q \cup \{\overrightarrow{x_2 < x_1}\}$.

12. $\overrightarrow{x_2 = x_3}$ $d_{x_2} = \{2\}$ & $d_{x_3} = \{2\}$

- $x_2 = 2 > 2$ ✓

Then, no changes.

13. $\overrightarrow{x_2 \neq x_4}$ $d_{x_2} = \{2\}$ & $d_{x_4} = \{3\}$

- $x_2 = 2 \neq 3$ ✓

Then, no changes.

14. $\overrightarrow{x_3 \neq x_4}$ $d_{x_3} = \{2\}$ & $d_{x_4} = \{3\}$

- $x_3 = 2 \neq 3$ ✓

Then, no changes.

15. $\overrightarrow{x_2 \neq x_4}$ $d_{x_2} = \{2\}$ & $d_{x_4} = \{3\}$

- $x_2 = 2 \neq 3$ ✓

Then, no changes.

16. $\overrightarrow{x_4 < x_5}$ $d_{x_4} = \{3\}$ & $d_{x_5} = \{4\}$

- $x_4 = 3 < 4$ ✓

Then, no changes.

17. $\overrightarrow{x_2 < x_1}$ $d_{x_2} = \{2\}$ & $d_{x_1} = \{3\}$

- $x_2 = 2 < 3$ ✓

Then, no changes. End of the algorithm.

We obtain the following values: $d_{x_1} = \{3\}$, $d_{x_2} = \{2\}$, $d_{x_3} = \{2\}$, $d_{x_4} = \{3\}$ and $d_{x_5} = \{4\}$. After running $AC-3$, no further changes (domain reductions) are needed, then the network has reached arc-consistency because arcs are consistent with the constraints. As a reminder, **arc-consistency** means that for each variable, for each possible value in its domain, there must be a valid value in the domain of the other variables that satisfies all constraints.

2.4 Exercise 4

The problem can be modeled as follows:

- **Variable X:** The variables are the positions of the marks on the ruler. If we have k marks, we define the variables as $x_1, \ldots, x_k$, where each variable x_i represents the position of the $i-th$ mark on the ruler. The positions must be integers.
- **Domain D:** We can consider the domain of each variable as a bounded interval of intervals with lower bound being 0 and upper bound a given M.
- **Constraints C:** The constraints are that the distance (absolute difference) between each pair of marks must be distincct. Specifically, for each pair x_i and x_j we want: $|x_i - x_j|$ to be different for every pair of marks, and this can be expressed as:

$$|x_i - x_j| \neq |x_m - x_n| \quad \forall i, j, m, n \text{ where } i \neq j, m \neq n.$$

Now, let's imagine that they give us just a certain number of marks k and we want to find the smallest Golomb Ruler, which would be the smallest of the largest marks. Therefore, what we would do is:

$$\text{Minimize } M = max\{x_1, \ldots, x_k\}$$
$$\text{Such that no 2 distances are the same}$$

The rest of the analysis can be found on the github folders.

2.5 Exercise 5

In corresponding folder.