1 Wooden Stick Grid Puzzle

Academic Integrity Statement

If your written submission on Learn does not include this academic integrity statement with your signature (typed name), we will deduct 5 marks from your final assignment mark.

I declare the following statements to be true:

- The work I submit here is entirely my own.
- I have not shared and will not share any of my code with anyone at any point.
- I have not posted and will not post my code on any public or private forum or website.
- I have not discussed and will not discuss the contents of this assessment with anyone at any point.
- I have not posted and will not post the contents of this assessment and its solutions on any public or private forum or website.
- I will not search for assessment solutions online.
- I am aware that misconduct related to assessments can result in significant penalties, possibly including failure in the course and suspension. This is covered in Policy 71: https://uwaterloo.ca/secretariat/policies-procedures-guidelines/policy-71.

By typing or writing my full legal name below, I confirm that I have read and understood the academic integrity statement above.

An Nguyen 20726636

	2	3	3	1	
2	3	2	1	4	1
1	4	1	3	2	3
3	2	3	4	1	2
2	1	4	2	3	2
	3	1	2	2	

Table 1: Wooden Stick Grid Puzzle Example 1

(b) Formulate the puzzle in Table 1 as a constraint satisfaction problem. We have provided the variable definitions below. Describe the domains of the variables and the constraints. Feel free to use words or mathematical expressions.

Variables:

A variable represents the sequence of four numbers from left to right in a row or the sequence of four numbers from top to bottom in a column.

Let R_i denotes the variable for the i-th row from the top. $i \in \{1, 2, 3, 4\}$ R_1 denotes the top row.

Let C_j denotes the variable for the j-th column from the left. $j \in \{1, 2, 3, 4\}$. C_1 denotes the leftmost column.

Marking Scheme: (8 marks)

- (2 marks) Domains
- (6 marks) Constraints

Next Page

(c) Consider the puzzle in Table 1.

For each unary constraint on a variable, remove all the values in the variable's domain that violate the unary constraint. Afterwards, write down the updated domains of all the variables.

- $R_1 \in \{3214, 3124\}$
- $R_2 \in \{4132, 4231, 4312\}$
- $R_3 \in \{1243, 1342, 2341\}$
- $R_4 \in \{1423, 2413, 3241, 3142, 3412, 2143\}$
- $C_1 \in \{3421, 2431, 1432\}$
- $C_2 \in \{2314, 1324, 2134\}$
- $C_3 \in \{1243, 1342, 2341\}$
- $C_4 \in \{4123, 4213\}$

1 b/ Domains: $R_i = [x_1, x_2, x_3, x_4]$ $i, x_1, x_2, x_3, x_4 \in \{1,2,3,4\}$

$C_j = [y_1, y_2, y_3, y_4]$ $j, y_1, y_2, y_3, y_4 \in \{1,2,3,4\}$

Constraints:

All Different Constraint:

For all $R_i = [x_1, x_2, x_3, x_4]$: $x_1 \neq x_2 \neq x_3 \neq x_4$

For all $C_j = [y_1, y_2, y_3, y_4]$ $y_1 \neq y_2 \neq y_3 \neq y_4$

Viewing Constraint:

Given View V_{pk} where p denotes the position of viewing, $p \in \{l, r, t, b\}$ (for left, right, top, bottom) respectively) and k denotes the row or collumn number (if $p = l$ or r, $k = i$, otherwise $k = j$)

The constraint is satisfied when:

Note: used varible def^n for row but can be subbed out for the collum def^n wlog

If $p = l$ or t:

$$V_{pk} = \begin{cases} 1 & x_1 = 4 \\ 2 & \exists\, x_a < x_b \wedge \nexists\, x_c < x_d < x_e \wedge x_1 \neq 4 \\ & \wedge\, a < b \wedge c < d < e \quad a, b, c, d, e \in \{1,2,3,4\} \\ 3 & \exists\, x_a < x_b < x_c \wedge x_1 \neq 4 \wedge R_k \neq [1,2,3,4] \\ & \wedge\, a < b < c \quad a, b, c \in \{1,2,3,4\} \\ 4 & R_k = [1,2,3,4] \end{cases}$$

Note: for (2) means there are two sticks (One shorter than the other) and constraint for (3) is not satisfied, nor is for (1)

for (3) means there ar 3 sticks in increasing order and constraint I is not satisfied.

If $p = r, b$:

$$V_{pk} = \begin{cases} 1 & x_4 = 4 \\ 2 & \exists\, x_a > x_b \wedge \nexists\, x_c > x_d > x_e \wedge x_4 \neq 4 \\ & \wedge\, a < b \wedge c < d < e \qquad a,b,c,d,e \in \{1,2,3,4\} \\ 3 & \exists\, x_a > x_b > x_c \wedge x_4 \neq 4 \wedge R_k \neq [4,3,2,1] \wedge a < b < c \\ & \qquad a,b,c \in \{1,2,3,4\} \\ 4 & R_k = [4,3,2,1] \end{cases}$$

Overlap constraint:

Consider Z_{ij} for cell at row i and collumn j.

$$\exists\, Z_{ij} = R_i[x_j] = C_j[y_i] \qquad i,j \in \{1,2,3,4\}$$

(collum and row values must match up for cell Z_{ij})

1 d/ (2) remove 3241, 3142, 3412 from R_4
Add nothing back to S

(3) remove 2431, 1432 from C_1
add back $\langle R_3(R_3,C_1)\rangle, \langle R_4(R_4,C_1)\rangle$

(4) remove 2413, 2143 from R_4
add back nothing

(5) remove 4123 from C_4
add back nothing

(6) remove 1243, 2341 from C_3
add back nothing

(7) remove 2314, 1342 from C_2
add back nothing

(8) remove 4231, 4312 from R_2
add back nothing

(9) remove 3124 from R_1
add back $\langle C_1(R_1,C_1)\rangle$

e/ The two methods are different in that by hand, you solve the puzzle cell by cell with the given constraints in mind wheras by AC-3, you generate all possible values and narrow down the solution using the binary constraints. The methods are similar in the order they consider the constraints. While I was solving it by hand I found that I would consider the unary constraint of all different then viewing constraints to consider what values a cell can have and then used the binary constraint of matching for that cell to determine the cell. This is the same process AC-3 uses to narrow down the possible results.

2 Decision Trees

b)

1. The maximum depth of the decision tree is 21 with root being depth 0.

2. The best maximum depth for pre-prunin is 7.

3. The best minimum number of examples for post-pruning is 60.

4. Choosing the full tree would be a bad strategy as it is likely overfitted to the data. That leaves the pruning using max-depth or min-examples. Of these two I would choose to go with max-depth as it would likely be smaller and easier to interpret as well as less expensive to make. By using a minimum number of examples, we must start with a full tree then take out nodes until the minimum number of examples is satisfied. This contrasts with the maximum depth where we would only generate nodes up until the maximum depth which is much faster. For this reason I would choose the maximum depth strategy.