Q1. CSP: Air Traffic Control

We have five planes: A, B, C, D, and E and two runways: international and domestic. We would like to schedule a time slot and runway for each aircraft to **either** land or take off. We have four time slots: $\{1, 2, 3, 4\}$ for each runway, during which we can schedule a landing or take off of a plane. We must find an assignment that meets the following constraints:

- Plane B has lost an engine and must land in time slot 1.
- Plane D can only arrive at the airport to land during or after time slot 3.
- Plane A is running low on fuel but can last until at most time slot 2.
- Plane D must land before plane C takes off, because some passengers must transfer from D to C.
- No two aircrafts can reserve the same time slot for the same runway.

(a) Complete the formulation of this problem as a CSP in terms of variables, domains, and constraints (both unary and binary). Constraints should be expressed implicitly using mathematical or logical notation rather than with words.

Variables: A, B, C, D, E for each plane. **Domains:** {1, 2, 3, 4} X {INTER, DOMES}

Constraints: (you do not have to use all lines)

B[1] = 1

D[1] >= 3

A[1] <= 2

D[1] < C[1]

A != B != C != D

(b) For the following subparts, we add the following two constraints:

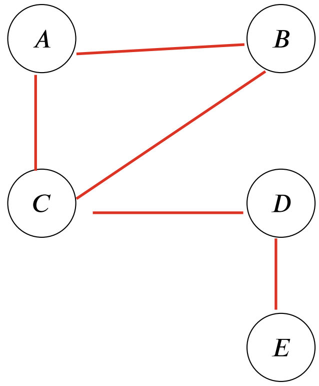
- Planes A, B, and C cater to international flights and can only use the international runway.
- Planes D and E cater to domestic flights and can only use the domestic runway.

A[2] = B[2] = C[2] = INTER ; D[2] = E[2] = DOMES

(i) With the addition of the two constraints above, we completely reformulate the CSP. You are given the variables and domains of the new formulation. Complete the constraint graph for this problem given the original constraints and the two added ones.

Variables: A, B, C, D, E for each plane. **Constraint Graph:**

Domains: $\{1, 2, 3, 4\}$



(ii) What are the domains of the variables after enforcing arc-consistency? Begin by enforcing unary constraints. (Cross out values that are no longer in the domain.)

A	$\cancel{1}$	2	$\cancel{3}$	$\cancel{4}$
B	1	$\cancel{2}$	$\cancel{3}$	$\cancel{4}$
C	$\cancel{1}$	$\cancel{2}$	$\cancel{3}$	4
D	$\cancel{1}$	$\cancel{2}$	3	$\cancel{4}$
E	1	2	$\cancel{3}$	4

(iii) Arc-consistency can be rather expensive to enforce, and we believe that we can obtain faster solutions using only **forward-checking** on our variable assignments. Using the Minimum Remaining Values heuristic, perform backtracking search on the graph, breaking ties by picking lower values and characters first. List the $(variable, assignment)$ pairs in the order they occur (including the assignments that are reverted upon reaching a dead end). Enforce unary constraints before starting the search.

(You don't have to use this table, it won't be graded.)

A	$\cancel{1}$	2	$\cancel{3}$	$\cancel{4}$
B	1	$\cancel{2}$	$\cancel{3}$	$\cancel{4}$
C	$\cancel{1}$	$\cancel{2}$	$\cancel{3}$	4
D	$\cancel{1}$	$\cancel{2}$	3	$\cancel{4}$
E	1	2	$\cancel{3}$	4

Answer:

(c) Suppose we have just one runway and n planes, where no two planes can use the runway at once. We are assured that the constraint graph will always be tree-structured and that a solution exists. What is the runtime complexity in terms of the number of planes, n, of a CSP solver that runs arc-consistency and then assigns variables in a topological ordering?

- ○ $O(1)$
- ○ $O(n)$
- ○ $O(n^2)$
- ● $O(n^3)$
- ○ $O(n^n)$
- ○ None of the Above

Q2. CSPs

In this question, you are trying to find a four-digit number satisfying the following conditions:

1. the number is odd,
2. the number only contains the digits 1, 2, 3, 4, and 5,
3. each digit (except the leftmost) is strictly larger than the digit to its left.

(a) CSPs

We will model this as a CSP where the variables are the four digits of our number, and the domains are the five digits we can choose from. The last variable only has 1, 3, and 5 in its domain since the number must be odd. The constraints are defined to reflect the third condition above. Thus before we start executing any algorithms, the domains are

1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 3 5

(i) Before assigning anything, enforce arc consistency. Write the values remaining in the domain of each variable after arc consistency is enforced.

1, 2	2, 3	3, 4	5

(ii) With the domains you wrote in the previous part, which variable will the MRV (Minimum Remaining Value) heuristic choose to assign a value to first? If there is a tie, choose the leftmost variable.

- ○ The first digit (leftmost)
- ○ The second digit
- ○ The third digit
- ● The fourth digit (rightmost)

(iii) Now suppose we assign to the leftmost digit first. Assuming we will continue filtering by enforcing arc consistency, which value will LCV (Least Constraining Value) choose to assign to the leftmost digit? **Break ties from large (5) to small (1).**

- [x] 1
- [ ] 2
- [ ] 3
- [ ] 4
- [ ] 5

(iv) Now suppose we are running min-conflicts to try to solve this CSP. If we start with the number 1332, what will our number be after one interation of min-conflicts? Break variable selection ties from left to right, and **break value selection ties from small (1) to large (5)**.

1232 ____________________

(b) The following questions are completely unrelated to the above parts. Assume for these following questions, there are only binary constraints unless otherwise specified.

(i) [*true* or *false*] When enforcing arc consistency in a CSP, the set of values which remain when the algorithm terminates does not depend on the order in which arcs are processed from the queue. True

(ii) [*true* or *false*] Once arc consistency is enforced as a pre-processing step, forward checking can be used during backtracking search to maintain arc consistency for all variables. False

(iii) In a general CSP with n variables, each taking d possible values, what is the worst case time complexity of enforcing arc consistency using the AC-3 method discussed in class?

○ 0 ○ $O(1)$ ○ $O(nd^2)$ ● $O(n^2d^3)$ ○ $O(d^n)$ ○ ∞

(iv) In a general CSP with n variables, each taking d possible values, what is the maximum number of times a backtracking search algorithm might have to backtrack (i.e. the number of the times it generates an assignment, partial or complete, that violates the constraints) before finding a solution or concluding that none exists?

○ 0 ○ $O(1)$ ○ $O(nd^2)$ ○ $O(n^2d^3)$ ● $O(d^n)$ ○ ∞

(v) What is the maximum number of times a backtracking search algorithm might have to backtrack in a general CSP, if it is running arc consistency and applying the MRV and LCV heuristics?

○ 0 ○ $O(1)$ ○ $O(nd^2)$ ○ $O(n^2d^3)$ ● $O(d^n)$ ○ ∞